PB82-242074

A Microprocessor ASCII Character Buffering System

(U.S.) Environmental Monitoring and Support Lab.
Cincinnati, OH

Jul 82

U.S. Department of Commerce
National Technical Information Service
NTIS

PB82-242074

EPA 600/4-82-051
July 1982

A MICROPROCESSOR ASCII CHARACTER BUFFERING SYSTEM

by

John M. Teuschler
Roger A. Jollis

Physical and Chemical Methods Branch
Environmental Monitoring and Support Laboratory

Jonathan E. Kopke

University of Cincinnati Computing Center

ENVIRONMENTAL MONITORING AND SUPPORT LABORATORY
OFFICE OF RESEARCH AND DEVELOPMENT
U.S. ENVIRONMENTAL PROTECTION AGENCY
CINCINNATI, OHIO 45268

REPRODUCED BY
NATIONAL TECHNICAL
INFORMATION SERVICE
U.S. DEPARTMENT OF COMMERCE

TECHNICAL REPORT DATA
(Please read Instructions on the reverse before completing)

1. REPORT NO.
EPA-600/4-82-051

2.
ORD Report

3. RECIPIENT'S ACCESSION NO.
PB82 242074

4. TITLE AND SUBTITLE
A Microprocessor ASCII Character Buffering System

5. REPORT DATE
~~6/81~~ July 1982

6. PERFORMING ORGANIZATION CODE

7. AUTHOR(S)
John M. Teuschler, Roger Jollis, Jonathan Kopke

8. PERFORMING ORGANIZATION REPORT NO.

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Environmental Monitoring and Support Laboratory
26 W. St. Clair Street
Cincinnati, Ohio 45268

10. PROGRAM ELEMENT NO.
ABEB1C

11. CONTRACT/GRANT NO.

12. SPONSORING AGENCY NAME AND ADDRESS
Environmental Monitoring and Support Laboratory
26 W. St. Clair Street
Cincinnati, Ohio 45268

13. TYPE OF REPORT AND PERIOD COVERED
Applications Guide 6/80-12/80

14. SPONSORING AGENCY CODE
EPA-600/06

15. SUPPLEMENTARY NOTES

16. ABSTRACT

A microprocessor buffering system (MBS) was developed at the Environmental Monitoring and Support Laboratory - Cincinnati (EMSL-CI) to provide an efficient transfer for serial ASCII information between intelligent instrument systems and a Data General NOVA laboratory automation computer system. Since the MBS is field programmable, the design is expected to satisfy many of the Agency's needs to transfer data from "intelligent instrument" computer systems to the standard EPA laboratory automation system. The system has the capability of transferring information simultaneously from three intelligent instrument systems.

17. KEY WORDS AND DOCUMENT ANALYSIS

a. DESCRIPTORS	b. IDENTIFIERS/OPEN ENDED TERMS	c. COSATI Field/Group

18. DISTRIBUTION STATEMENT
Release to public

19. SECURITY CLASS *(This Report)*
Unclassified

20. SECURITY CLASS *(This page)*
Unclassified

21. NO. OF PAGES

22. PRICE

EPA Form 2220-1 (9-73)

DISCLAIMER

This report has been reviewed by the Environmental Monitoring and Support Laboratory - Cincinnati, U.S. Environmental Protection Agency, and approved for publication. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

EPA Form 2350-4 (4-80)

TYPING GUIDE SHEET

FOREWORD

Environmental measurements are required to determine the quality of ambient waters and the character of waste effluents. The Environmental Monitoring and Support Laboratory - Cincinnati, conducts research to:

- Develop and evaluate techniques to measure the presence and concentration of physical, chemical, and radiological pollutants in water, wastewater, bottom sediments, and solid waste.

- Investigate methods for the concentration, recovery, and identification of viruses, bacteria and other microbiological organisms in water; and to determine the responses of aquatic organisms to water quality.

- Develop and operate an Agency-wide quality assurance program to assure standardization and quality control of systems for monitoring water and wastewater.

- Develop and operate a computerized system for instrument automation leading to improved data collection, analysis, and quality control.

This report was developed in the Advanced Instrumentation Section of the Environmental Monitoring and Support Laboratory (EMSL) in the interest of advancing laboratory techniques and quality control through computerization.

Robert L. Booth

Robert L. Booth, Acting Director
Environmental Monitoring and Support
Laboratory - Cincinnati

ABSTRACT

A microprocessor buffering system (MBS) was developed at the Environmental Monitoring and Support Laboratory - Cincinnati (EMSL-CI) to provide an efficient transfer of serial ASCII information between intelligent instrument systems and a Data General NOVA laboratory automation computer system.[1] Since the MBS is field progammable, the design is expected to satisfy many of the Agency's needs to transfer data from "intelligent instrument" computer systems to the standard EPA laboratory automation system. The system has the capability of transferring information simultaneously from three intelligent instrument systems.

The microprocessor buffer system is programmed to store in random access memory any "software flagged" serial data appearing on any one of three universal asynchronous receiver/transmitter (UART) ports. The data is collected until a pre-programmed software flag is received. The cease in data collection triggers the MBS to log into the NOVA system as a BASIC user. A BASIC program is then used by the NOVA to accept the data from the MBS and transfer it to a disk file. Once the data has been completely transferred, the MBS logs off the NOVA system, thereby freeing the port for another BASIC user. This microprocessor mode of transfer completely eliminates any manual intervention. With the data on the NOVA system, the "intelligent instrument" operator can then access the transferred data through the multi-user BASIC terminals and do further data reduction/manipulation on the transferred data in a high level language.

Once the data is on the NOVA system, processed data can be directly transmitted to data management/sample tracking computer systems or archived on magnetic tape.

EPA Form 2350-4 (4-80)
(PREVIOUSLY CIN. EPA FORM 257)
TYPING GUIDE SHEET

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD, BEGIN SECTIONS HERE

CONTENTS

Page

Foreword...iii
Abstract... iv
Figures...vii
1. Introduction... 1
2. Background... 2
3. System Objectives... 4
4. Microprocessor Buffer System - Functional Overview... 5
    4.1 Real Time Data Acquisition... 5
    4.2 Data Discrimination... 5
    4.3 Data Storage... 7
    4.4 Data Transmission... 7
        4.4.1 Logon... 8
        4.4.2 Data Transmission Error Conditions... 8
        4.4.3 Logoff... 9
5. Hardware Implementation... 10
    5.1 System Hardware - Mainframe and CPU... 10
    5.2 Communications... 10
    5.3 Memory... 11
    5.4 Diagnostic Terminal... 11
    5.5 Custom Hardware... 12
6. Software Implemetation... 18
    6.1 Non-Interrupt Driven Software... 18
    6.2 Real-Time Software... 21
7. Program Documentation... 24
    7.1 SETUP... 24
    7.2 TB534... 48
    7.3 WAIT... 53
    7.4 OPER... 74
    7.5 USER... 75
    7.6 NOVAST and NOVA... 76
8. Notes to the System Manager...123
    8.1 Connecting the Microprocessor...123
    8.2 Booting the System...123
    8.3 The System Status Report on the System Console...124
    8.4 NOVA Directories, Login Messages, and BASIC ID Requirements..126
    8.5 The BASIC Program in the NOVA...126
9. References...135

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA, OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

PAGE NUMBER
EPA-287 (Cin.)
(4-76)

CONTENTS (Cont'd.)

Appendices

A. Jumper Configurations for the SBC 80/20-4 Board....................129
B. Jumper Configurations for the iSBC-534 Board.......................130
C. Variables Located in the Utility Storage on the SBC 80/20-4 Board.131
D. Microprocessor Memory Allocation...................................134

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA

FIGURES

1. EPA Laboratory Automation System.. 3
2. Microprocessor Buffer System... 6
3. MBS Status Display on the Hewlett-Packard Terminal....................... 13
4. Monitor Interface Front Panel Display...................................... 14
5. Block Diagram of the Monitor Interface..................................... 15
6. Schematic of the Monitor Interface... 17
7. Block Diagram of the Non-Interrupt Driven Software....................... 19
8. Block Diagram of the Interrupt Driven Software............................ 22
9. SETUP Flowchart.. 27
10. TB534 Flowchart... 49
11. Configuration of the User Status Word..................................... 54
12. Configuration of the System Status Word................................... 54
13. WAIT Flowchart.. 55
14. OPER Flowchart.. 59
15. Memory Block Configuration.. 68
16. USER Flowchart (a).. 71
17. USER Flowchart (b).. 72
18. FRESH Flowchart... 73
19. NOVAST Flowchart.. 94
20. NOVA Flowchart (a).. 95
21. NOVA Flowchart (b).. 96
22. Booting Procedure Prompts...125
23. System Console Status Information...125
24. Rudimentary Micro Input Program...127

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA. OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS.

PAGE NUMBER

SECTION 1

INTRODUCTION

A modern analytical laboratory's computer power is most often a conglomerate of a large scale, multi-language, multi-user, main-frame computer (either local or remote) used for extensive computations, and a myriad of smaller instrument or multi-instrument computer systems used for real-time data acquisition and control. Instrument computer systems are usually supplied by manufacturers with packaged software. In many instances, it is highly desirable to transfer the data from the intelligent instrument systems to larger, more powerful computers for further data processing and/or insertion into data management/sample tracking systems.

Since intelligent instrument systems usually lack sufficient computer communication protocol (that is, EIA-RS-232C interfaces normally lack "clear to send" and "request to send" signals, the transfer of data from these computer systems to a main-frame computer is plagued with incompatabilities. There is no handshaking from the receiver to acknowledge valid reception of data from the instrument system; nor is there any acknowledgement to the instrument system that the receiver is ready for the data to be transmitted. This presents a situation which is prohibitive to valid data transmission unless special procedures are adopted.

One such special procedure requires the allocation of a memory buffering mechanism to accept information from intelligent instruments. On a multi-user system, this would require the dedication of a block of memory and a number of dedicated ports to function as the buffering mechanism. This procedure presents an extreme burden to any system having multiple users and limited memory.

Another possible procedure would be to change the manufacturer's system programming to accommodate a communication protocol conducive to valid data transmission. However, changing a manufacturer's system programming is frequently difficult or cost-prohibitive.

Since changing the manufacturer's system is impractical, the remaining approach is to buffer the information. As discussed above, it would be inefficient to burden the main computer with the task of buffering the information. Therefore, it is reasonable to distribute the buffering of the data to an eternal buffering system. As a result, the objective of this project was to design and build a general purpose system, utilizing commercially available hardware that could be applied to moving data from a variety of instrument computer systems to EPA's laboratory automation computer system.

SECTION 2

BACKGROUND

Figure 1 illustrates an early 1970's lab automation system used to automate several EPA environmental monitoring laboratories.[1] This medium scale integration Data General NOVA computer system was designed to collect data directly from the instruments utilizing Data General's multi-user, extended BASIC language, with calls to assembly language subroutines. Atomic absorption units, continuous flow analyzers, ultraviolet-visible spectrophotometers, emission spectrometers, and electronic balances are among the instrumentation interfaced to the system. The system has been used since 1975 to collect real-time data, process the raw data in extended BASIC, and produce final output reports.

Furthermore, the laboratory automation system has the capability of being included in a hierarchical computer network through the synchronous line adapter (SLA) shown in Figure 1. Thus, laboratory data can be transferred to larger data base management computers in general, and to the EPA Sample File Control System specifically. Standard software is purchased to drive the SLA hardware. The Houston Attached Support Processor (HASP) software or 2780 emulator software is available for use on the NOVA. The NOVA system was specified with two programming grounds. This allows multi-user BASIC, which interfaces to the instrumentation, to run in the foreground, while background is used for system functions and communications, thereby supplying the link to the next computer. The system offers a relatively easy to use, high level language for the processing of labortory data.

Users of commercial instrument systems, such as the Jarrell-Ash inductively coupled argon plasma emission spectrometer and the Varian gas chromatography data system, recognized the unique capabilities of the NOVA laboratory automation system and desired a data link from their systems to the NOVA. Primordial attempts at such a data link usually consisted of carrying punched paper tape to the NOVA. Besides being cumbersome for the operator, the approach was slow and the system console had to be utilized to read the tapes and transfer the data to a disk file. A better solution was needed.

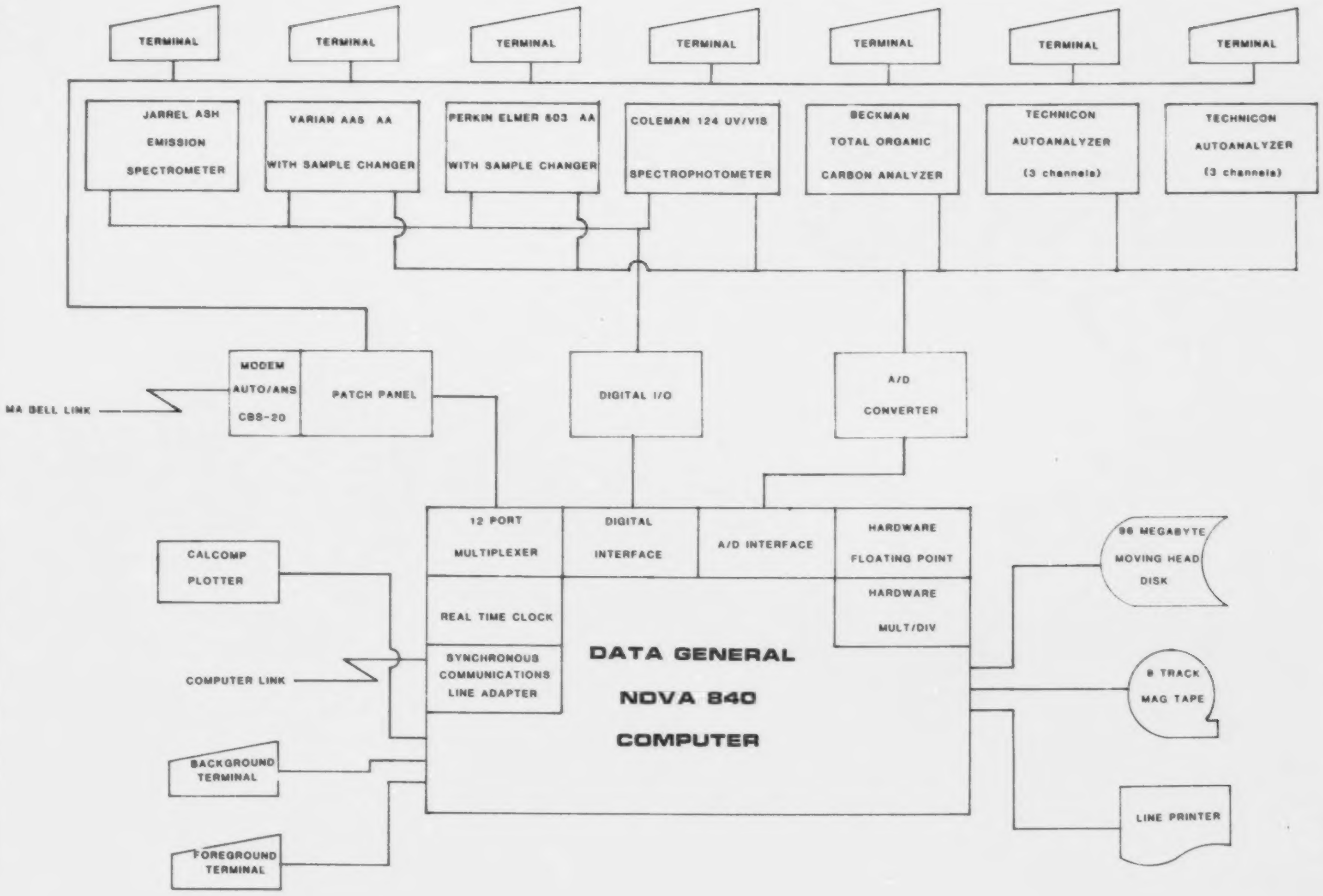

TERMINAL
TERMINAL
TERMINAL
TERMINAL
TERMINAL
TERMINAL
TERMINAL
JARREL ASH EMISSION SPECTROMETER
VARIAN AAS AA WITH SAMPLE CHANGER
PERKIN ELMER 503 AA WITH SAMPLE CHANGER
COLEMAN 124 UV/VIS SPECTROPHOTOMETER
BECKMAN TOTAL ORGANIC CARBON ANALYZER
TECHNICON AUTOANALYZER (3 channels)
TECHNICON AUTOANALYZER (3 channels)
MA BELL LINK
MODEM AUTO/ANS CBS-20
PATCH PANEL
DIGITAL I/O
A/D CONVERTER
12 PORT MULTIPLEXER
DIGITAL INTERFACE
A/D INTERFACE
HARDWARE FLOATING POINT
HARDWARE MULT/DIV
REAL TIME CLOCK
SYNCHRONOUS COMMUNICATIONS LINE ADAPTER
DATA GENERAL NOVA 840 COMPUTER
CALCOMP PLOTTER
COMPUTER LINK
BACKGROUND TERMINAL
FOREGROUND TERMINAL
96 MEGABYTE MOVING HEAD DISK
9 TRACK MAG TAPE
LINE PRINTER

FIGURE 1. EPA LABORATORY AUTOMATION SYSTEM

SECTION 3

SYSTEM OBJECTIVES

The goal of this project was to develop a microprocessor buffering system which would be able to transfer data from many commercial intelligent instrument systems to a Data General NOVA computer operating under the multi-user Extended BASIC, Revision 4.3. The microprocessor system had to provide:

- field programmability to meet the needs of various laboratories;
- a communication protocol for valid data reception and transmission;
- an expandable buffering area;
- system implementation utilizing commercially available hardware;
- bidirectional communication between the intelligent instrument operator and the microcomputer to alert the operator of the status; and
- simultaneous and spontaneous reception of ASCII information from multiple instruments.

SECTION 4

MICROPROCESSOR BUFFER SYSTEM (MBS)

The MBS was designed to function as a front end processor for the NOVA. Data is buffered in the MBS and retransmitted to the NOVA via a BASIC user port. As such, the MBS must provide real-time data acquisition for processors attempting to communicate with the NOVA. It must also provide random access memory (RAM) for the buffering of the information to the NOVA system, and read only memory (ROM) to contain the programming instructions.

Figure 2 is a block diagram of the MBS. Real-time data acquisition is accomplished by the control and manipulation of universal asynchronous receiver/transmitters (UARTS). A UART is a device which is used for data transmission. Parallel data is converted to serial data by the UART and is transmitted down a line. Likewise, serial data from another system is received by the UART (a character at a time), converted to parallel data, and placed on the bus of a computer.

4.1 REAL-TIME ACQUISITION

Each instrument has a dedicated port (UART) hardwired to the output of the intelligent instrument's micro/minicomputer. Immediately upon reception of an ASCII character, the UART interrupts the MBS so that the character can be processed. If the character is to be stored, it is written into the memory of the MBS.

The MBS can receive simultaneous data from the three intelligent instruments at individual data rates of up to 1200 baud. Every character generated by the intelligent instruments is received, processed, ignored, or stored, if the action is warranted. The MBS continuously monitors the outputs of the intelligent instruments through the incorporation of the dedicated interrupt generating UARTs.

4.2 DATA DISCRIMINATION

Not all data is to be buffered and transmitted to the NOVA. Since the MBS monitors all transmissions from the intelligent instruments, a special pre-programmed sequence of ASCII characters is used to start and stop the data buffering action. At system initialization, the MBS is programmed "on-site" for individual starting and stopping characters for each of the three intelligent instruments. A sequence of five characters is utilized as the triggering mechanism for the start-buffering and stop-buffering action. Likewise, the MBS is programmed "on-site" so that each intelligent

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
TOP OF IMAGE AREA
DROPPED HEAD. BEGIN SECTIONS HERE

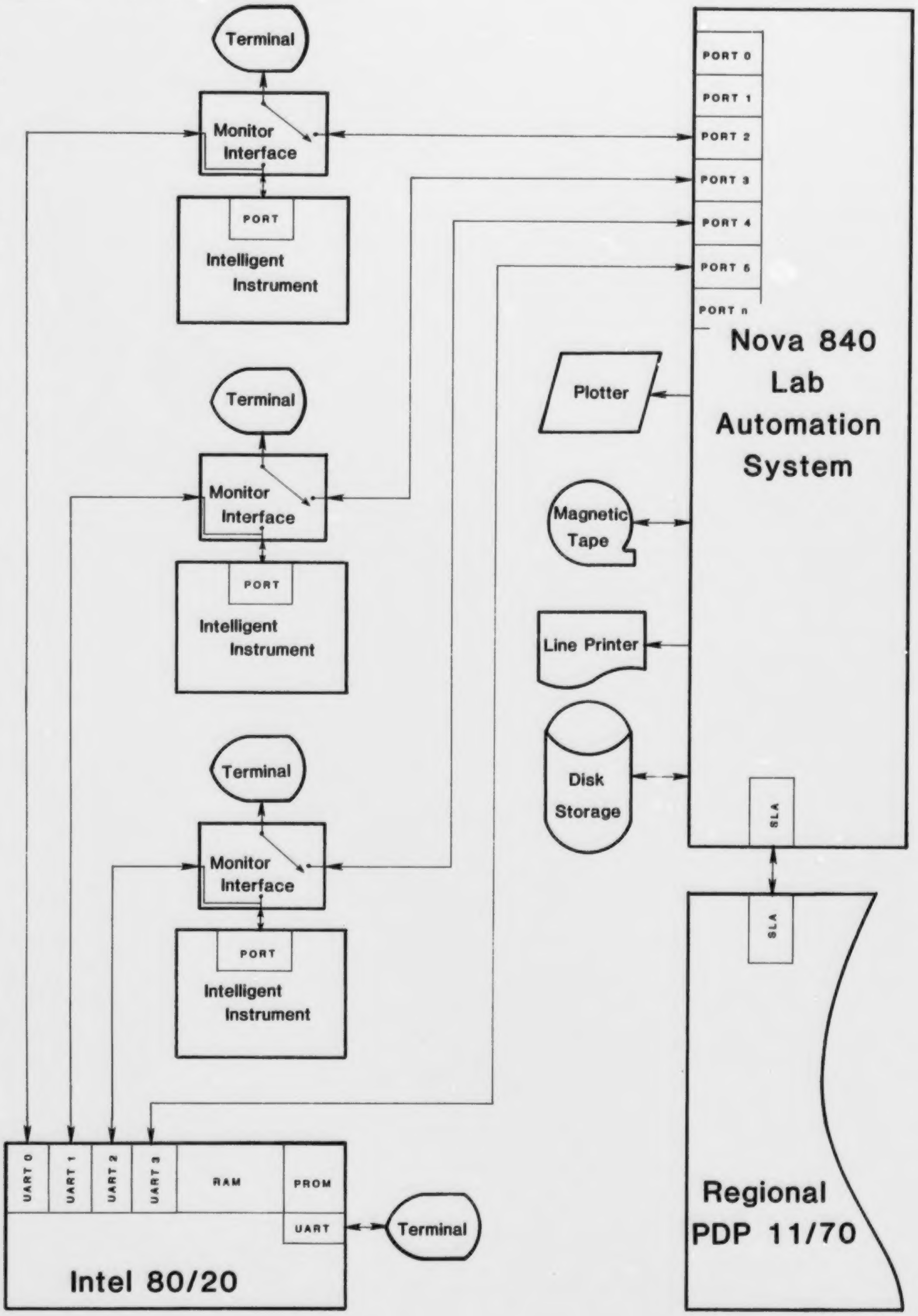

Terminal
Monitor Interface
PORT
Intelligent Instrument
Terminal
Monitor Interface
PORT
Intelligent Instrument
Terminal
Monitor Interface
PORT
Intelligent Instrument
UART 0
UART 1
UART 2
UART 3
RAM
PROM
UART
Terminal
Intel 80/20
PORT 0
PORT 1
PORT 2
PORT 3
PORT 4
PORT 5
PORT n
Nova 840 Lab Automation System
Plotter
Magnetic Tape
Line Printer
Disk Storage
SLA
SLA
Regional PDP 11/70

FIGURE 2. MICROPROCESSOR BUFFER SYSTEM.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA: OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"

EPA-287 (Cin.)
(4-76)

instrument has a particular directory on the NOVA system into which the data will be transferred.

Software discrimination works well for normal circumstances. However, during testing and start-up procedures for the intelligent instruments, the five starting characters may inadvertently be output. Therefore, a hardware switching function was included on the interface boxes at the intelligent instrument sites to allow the operator to disconnect the instrument. Therefore, the operator can disable the transmission of ASCII characters to the MBS, make several test injections to ascertain instrument validity, and then enable the transmission, thereby allowing valid information to be buffered by the MBS.

4.3 DATA STORAGE

The MBS can be configured with up to 48,000 bytes of memory buffer area. The system will automatically determine the size of its own memory and divide it into 1000-byte blocks. A dynamic memory allocation scheme is used to select blocks of available memory for buffering data. The memory blocks are used as they are needed. As the buffered data is transferred to the NOVA, the blocks are again placed in the free block pool. If the last free block of memory is used, a signal is sent to the operator to alert him of the situation. A hardware signal is also sent to the monitor interface boxes (see Figure 2) to prevent any further transmissions from the intelligent instruments until the problem is resolved.

The Data can be stored indefinitely in the MBS. Data can only be removed from the MBS by an organized transfer of the information to the NOVA or a complete MBS re-initialization. The MBS buffers sets of information. A set begins with the five starting characters and includes all the following ASCII characters.
The set terminates with the carriage return following the five stopping characters.

A set of information uses at least one memory block. The total number of memory blocks used for a set is only limited by the memory buffer size of the MBS. Similarly, the memory size is the only limitation on the number of sets of information contained in the memory buffer. A set of information is not transmitted to the NOVA until the complete set is received (starting, stopping and terminating characters).

4.4 DATA TRANSMISSION

The MBS was designed to mimic a human user and enter the data through a BASIC language user port. This mode of transferring the data through a high level language user port placed the least burden on the NOVA system.

The choice of a BASIC language program in the NOVA to receive communications from the microprocessor buffer was made due to the relative

EPA Form 2350-4 (4-80)
(PREVIOUSLY CIN: EPA FORM 283)
TYPING GUIDE SHEET

stability of the high level programming language. Data General and other manufacturers are constantly revising and improving the operating system software. This continuous state of flux makes it extremely difficult to utilize assembly code modules to interface the instrument systems and analog instruments. Therefore, this interface was designed in a manner which does not rely on the revision stability of Data General software. By interfacing at a BASIC user port, software maintenance between operating revisions will be minimized.

4.4.1 Logon

Upon reception of a complete set of information, a NOVA logon task is activated in the MBS. UART #3 will send an "escape" character to the NOVA. The port on the NOVA will respond according to the present activity on the port. The MBS system has been programmed to handle any possible combination of responses the NOVA can generate.

Since the logon task is activated by the complete reception of a set of data, the set of data has a particular account identification associated with it. Part of the logon task is to correctly log into the appropriate account to allow the transfer of information into the proper NOVA directory. Any problems in logon will cause an error status light to be activated at the monitor interface boxes as well as the MBS console. This will alert an operator to a potential data transmission problem. As long as remedial action is taken upon observance of the error status light, the MBS should never be in a situation where the memory of the system has been completely filled.

4.4.2 Data Transmission Error Conditions

Since the MBS is transferring information via the BASIC language, as a character is transmitted to the NOVA, the NOVA responds with an echo. The echo from the NOVA is compared by the MBS to the character which was sent; and if the two characters agree, the next character is sent. If for some reason there was a data transmission problem and the wrong character was echoed, the MBS transmits a "delete" character. Basic responds with an underscore character upon reception of a delete character. The MBS system will again attempt to transmit the original character. The design described above prevents erroneous characters from being stored in the NOVA.

If for some reason there is no response from the NOVA in a reasonable time, the MBS will terminate any further character transmissions and start over with the logon procedure. The MBS will again attempt to transfer the entire set of information to the NOVA.

Characters which cause problems with the BASIC language are pre-filtered and transmitted as their ASCII decimal equivalents (See Appendix E, Data General Extended BASIC User's Manual, Rev. 06). Commas, spaces, and escape characters are among the characters which are transformed into their decimal equivalents before transmission to the NOVA.

4.4.3 Log Off

Upon complete transmission of a set of data to the NOVA, the MBS will send an escape character to terminate the data transfer program (see Section 8.5). When the NOVA responds with an asterisk (this signals that BASIC is in the command mode), the MBS will bye out from the NOVA, thereby freeing the NOVA port for another foreground BASIC user.

4.5 SYSTEM STATUS

The console CRT displays the status of each intelligent instrument and of the MBS system as a whole, as shown in Figure 2. An intelligent instrument system may be ready, receiving information, transmitting information, or in the process of holding a complete set of information in the memory buffer (standby). Error conditions are also indicated on the system console. See Section 8.3 for more detailed information.

The monitor interface box at the intelligent instrument site also indicates the status of the system for the individual site. An operator should observe the status indicator on the monitor interface change from "ready" to "receiving" when the instrument system transmits the five character starting sequence to the MBS. The monitor interface will remain in this state until the five stopping characters are received along with the terminating carriage return. At this point, the status of the monitor interface will indicate "standby." The "ready" indicator will also appear indicating the MBS is again ready for another set of information. The "transmitting" status indicator on the monitor interface will be active only when the MBS is in the process of sending the set of information to the NOVA. Note that it is possible to have the monitor interface with several status indicators active at the same time. See Section 5.4 for more detailed information.

SECTION 5

HARDWARE IMPLEMENTATION

Intel Corporation hardware was chosen for the implementation of the MBS because it was exceptionally strong in both software and hardware support. Since software development is a major consideration of any microprocessor project, the software tools available on the MDS system significantly biased the selection of the hardware vendor. Intel Corporation also provided "off the shelf" hardware modules for an assortment of hardware functions. The selection of Intel products satisfied the "commercial availability" aspect of the project objectives.

5.1 SYSTEM HARDWARE (MAINFRAME AND CPU)

The specific hardware for the project consists of an SBC 80/20-4 system. The system is a four-slot chassis with eight-level interrupt capability. The four slots are occupied with a central processing unit (CPU) card, a communications card, and additional random access memory (RAM) cards. The system is available in a standard 19-inch rack mount chassis, which allows the MBS to be mounted in the 19-inch rack mounts containing the Data General NOVA equipment. The CPU card for the system was a SBC 80/20-4. The board has the capacity for up to 8192 bytes of read only memory (ROM), and 4096 bytes of random access memory (RAM). Jumper configurations for the CPU card can be obtained in Appendix A.

The programmable asynchronous RS232C serial interface on the board functions as the connection to the initializing and diagnostic console. The CPU card has the eight-level programmable interrupt control capability necessary to accommodate the real-time data acquisition fucntion, which receives simultaneous serial ASCII transmissions from several intelligent instruments. More detailed information can be obtained from the "SBC 80/20-4 Microcomputer Hardware Reference Manual" available from Intel Corporation.

5.2 COMMUNICATIONS

A four-channel communication expansion board (iSBC 534) was selected as the means to get serial ASCII data into the microprocessor system. The board has individual software programmable baud rate generation for each serial input/output (I/O) channel. This feature allows the field selection of communication rates to meet the varying baud rate needs of the instruments. When the system is initialized in the field, the operator has the option of selecting the individual baud rates of the three incoming ASCII character transmissions from the instrument systems. The operator can also select the baud rate the microcomputer will use to communicate the information to the NOVA.

The SBC 534 card has the capability of operating with priority encoded and programmable interrupt algorithms. This feature allows the 534 card to interrupt the CPU whenever a character has been received from any of the intelligent instruments. This feature is accomplished by strapping the 534 card (see Appendix B) to allow an interrupt to take place whenever there is a "received character" flag on any of the three UARTs continuously monitoring the intelligent instruments. To prevent one intelligent instrument from monopolizing the MBS, the three intelligent instruments are placed in an auto-rotating interrupt arrangement. This arrangement gives equal priority to all interruptors. Each interruptor, after receiving service, becomes the lowest priority level until the next interrupt occurs. This rotating priority mode ensures the processing of all the characters as they are received by the UARTs. Three of the four UARTs on the SBC 534 card are used to continuously monitor the intelligent instruments. The fourth UART is used to communicate with the NOVA. For more detailed information on the communications card, consult the "iSBC534 Four Channel Communications Board Hardware Reference Manual" from Intel Corporation.

5.3 MEMORY

The CPU card has the capability of holding 8192 bytes of programmable read only memory (PROM). Of the 8192 bytes, 5262 bytes were used for the microprocessor buffer system. There is also an additional 4096 bytes of RAM memory resident on the CPU card. The CPU RAM was used for the storage of the stack, the interrupt jump table, and utility storage for the microprocessor buffer system (see Appendix C). The CPU card contained all the needed memory resources necessary for system operation. Therefore, the additional expansion memory boards which were plugged into the remaining two slots of the SBC 80/20-4 system are used for storage of buffered data.

The memory used for buffering data is blocked into 1000-byte blocks at system initialization time. Therefore, a single 16K byte RAM card can be used for buffer memory, or a combination of a 16K byte and a 32K byte memory card can be used together. The above combination will give 48K bytes of storage for the buffered data. Since the system is auto-sizing, the amount of memory to be used to buffer the data is automatically found and blocked at system initialization. The memory allocation for the entire MBS is shown in Appendix D.

5.4 DIAGNOSTIC TERMINAL

A Hewlett-Packard 2640 terminal was utilized for the "on line" display of the communication buffering system. The status of the system is displayed on the terminal in real-time. This particular terminal was selected primarily because it is common in EPA laboratories. An example of the status display is shown in Figure 3. The display dynamically depicts the status of User 0, User 1 and User 2. The users can be active in any of four states. Referring to Figure 3, the "uP READY" mode corresponds to a condition that exists when the MBS is still waiting to receive the five starting character sequence for the particular user. "uP RCVG" mode is active when the associated user is sending a set of data to the MBS. When the five stopping characters and the terminating carriage return are received, the status changes to "uP STBY" signifying that a set of data has been buffered and is awaiting transfer to the NOVA system. If the system is presently transferring information to the NOVA, the "uP TRANS" mode will be active. The only two mutually exclusive modes are the uP READY and the uP RCVG. All other modes may be displayed in combination with each other.

The particular display shown in Figure 3 has a present status of:

- User 0 is waiting to receive the starting characters,
- User 1 is waiting to receive the starting characters, and a set of data is awaiting transmission to the NOVA.
- User 2 is in the midst of receiving a set of data, and a previously stored report is presently being transmitted to the NOVA.

Besides displaying the status of the users, the display gives an indication as to the overall status of the MBS. The message "SYSTEM RUNNING NORMALLY" will change to reflect any abnormal condition the MBS encounters. A audible alarm will also be sounded to alert an operator in the computer room. Typical abnormal displays include "NOVA NOT RESPONDING," and "MAX USERS OR ALL LINES BUSY." The operator is thereby made aware of the problem and may take corrective action if it is necessary.

The memory buffer size is also displayed on the console. In Figure 3 it is shown as 48 blocks. This corresponds to a 48K byte memory buffer size. A "blocks in use" count is also displayed to give the operator an idea of the amount of memory typically in use during the normal operation of the system.

5.5 CUSTOM HARDWARE

Communication between the microprocessor buffer system and the intelligent instrument is void of human intervention, however, a monitor interface box was designed to allow the intelligent instrument operator to be aware of the status of the communications. The MBS automatically updates status lights on this box at two-second intervals in the quiescent state.

The conditions displayed are shown in Figure 4. Figure 5 is a block diagram of the monitor interface box. Under the STATUS DISPLAY section of the interface, there are six separate functions which can be independently displayed: ready, receiving, standby, transmitting, error, and stop data. The ready function indicates that the buffer system is ready to receive the starting characters which initialize the buffering action. The receiving

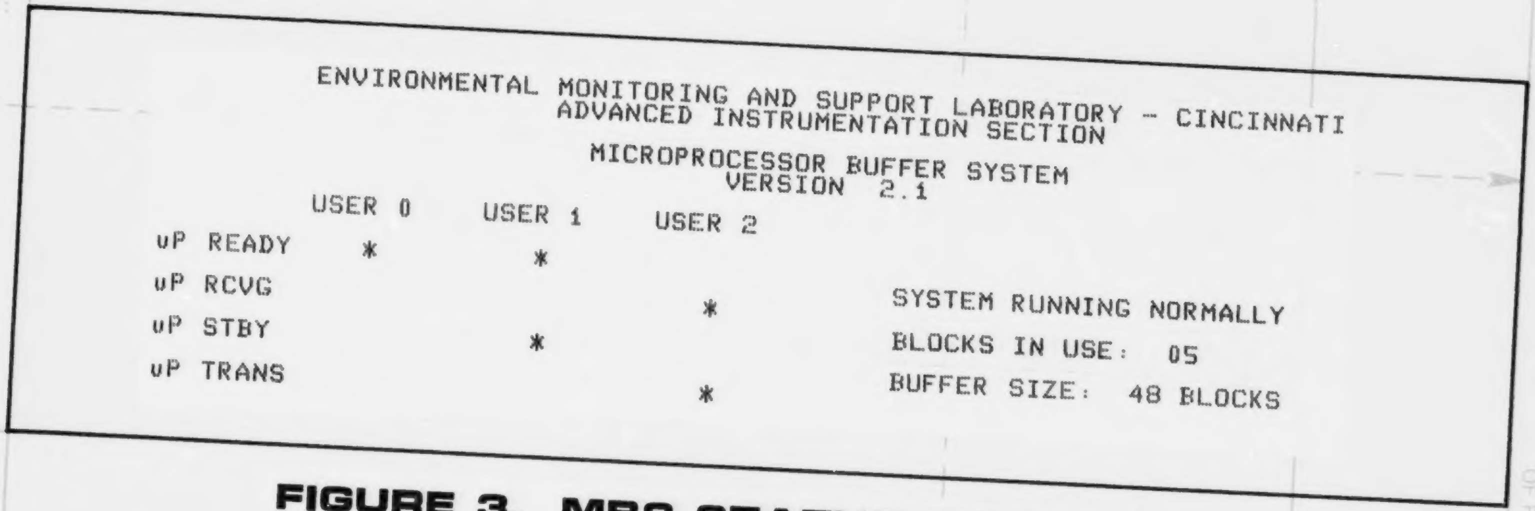
TYPING GUIDE SHEET

BEGIN FIRST LINE OF TEXT HERE

DROPPED HEAD: BEGIN SECTIONS HERE

CENTER OF PAGE

TOP OF IMAGE AREA



```
        ENVIRONMENTAL MONITORING AND SUPPORT LABORATORY - CINCINNATI
                       ADVANCED INSTRUMENTATION SECTION

                          MICROPROCESSOR BUFFER SYSTEM
                                 VERSION  2.1
            USER 0    USER 1    USER 2
 uP READY     *         *
 uP RCVG                                     SYSTEM RUNNING NORMALLY
                                  *
 uP STBY                *                    BLOCKS IN USE:  05
 uP TRANS                                    BUFFER SIZE:  48 BLOCKS
                                  *
```

FIGURE 3. MBS STATUS DISPLAY.

BEGIN LAST LINE OF TEXT

3/8"

BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

EPA-287 (Cin.)
(4-70)

PAGE NUMBER

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
DROPPED HEAD BEGIN SECTIONS HERE
TOP OF IMAGE AREA
BEGIN LAST LINE OF TEXT
3/8"
BOTTOM OF IMAGE AREA OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

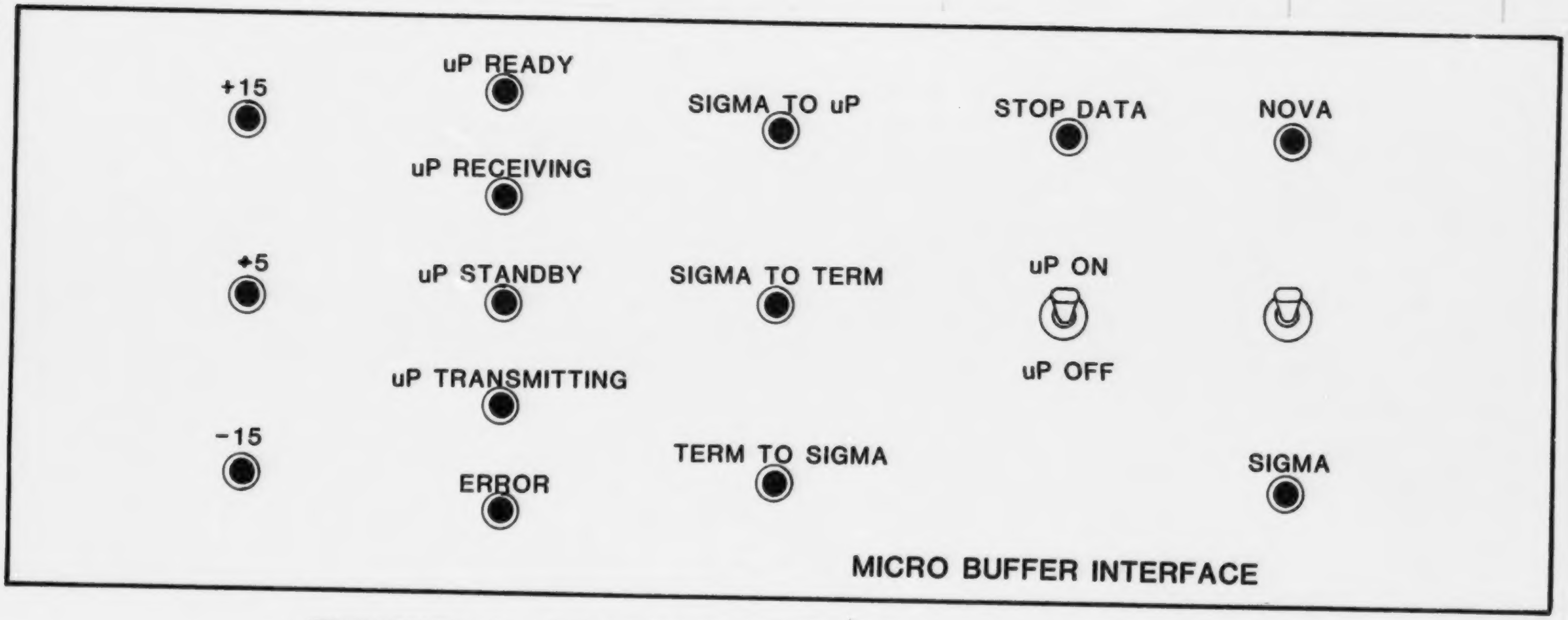

+15
+5
-15
uP READY
uP RECEIVING
uP STANDBY
uP TRANSMITTING
ERROR
SIGMA TO uP
SIGMA TO TERM
TERM TO SIGMA
STOP DATA
uP ON
uP OFF
NOVA
SIGMA
MICRO BUFFER INTERFACE

FIGURE 4. FRONT PANEL OF MONITOR INTERFACE.

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
DROPPED HEAD; BEGIN SECTIONS HERE
CENTER
TOP OF IMAGE AREA

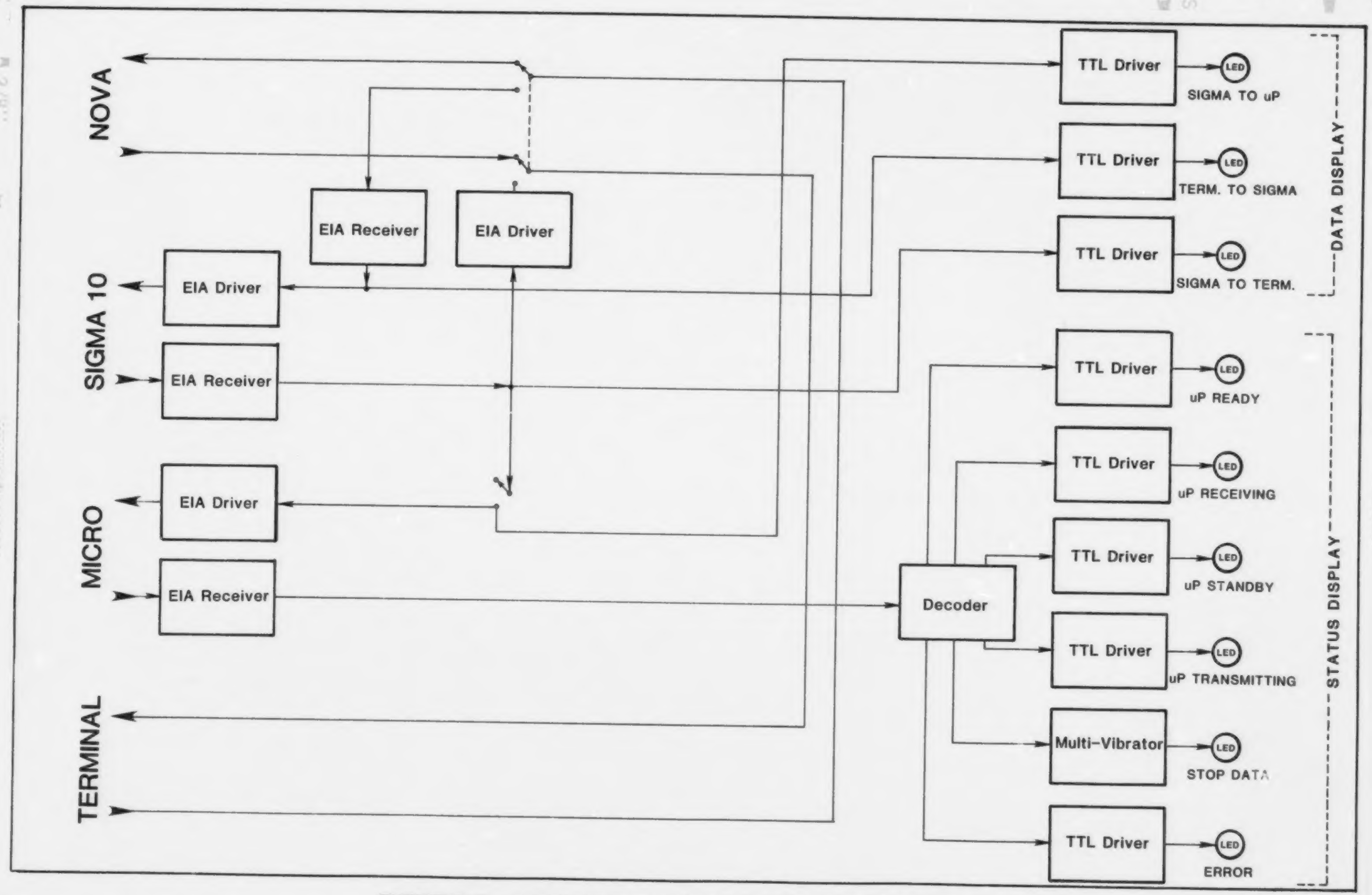

NOVA
SIGMA 10
MICRO
TERMINAL
EIA Receiver
EIA Driver
EIA Driver
EIA Receiver
EIA Driver
EIA Receiver
Decoder
TTL Driver
SIGMA TO uP
TTL Driver
TERM. TO SIGMA
TTL Driver
SIGMA TO TERM.
DATA DISPLAY
TTL Driver
uP READY
TTL Driver
uP RECEIVING
TTL Driver
uP STANDBY
TTL Driver
uP TRANSMITTING
Multi-Vibrator
STOP DATA
TTL Driver
ERROR
STATUS DISPLAY
LED

FIGURE 5. MONITOR INTERFACE BLOCK DIAGRAM.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA, OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"
PAGE NUMBER

EPA-287 (Cin.)
(4-76)

function indicates that the buffer system has received the starting characters and is now buffering the information in the RAM of the buffer system. The standby function signifies that a total report has been buffered. That is, the starting characters, stopping characters, and terminating carriage return have been received, and the information contained in the set has been stored in memory and is awaiting transfer to the NOVA. The transmitting function is activated when the buffer system has established communications with the NOVA and is proceeding with the transfer of information. The error function will occur if there is any problem with establishing communications with the NOVA. If this display occurs, the operator should immediately contact the system manager to ascertain the cause of the problem. The stop data function occurs when there is no longer a free block of RAM memory. This function signals the intelligent instrument operator to stop any further transmissions of information. The stop data function only occurs if all the RAM memory has been filled. The function not only displays the information but also drops the "request to send" flag going to the intelligent instrument. In instruments where the feature is installed and operational, this flag prevents any further transmissions from emanating from the intelligent instrument.

As can be seen from the block diagram, the status display is a decoded EIA-RS-232C transmission coming from the buffer system. Also, the data display section of the interface indicates a transfer of information between the intelligent instrument, the terminal, and the buffer system.

A switch was incorporated into the interface to allow the terminal to communicate with the intelligent instrument or with the NOVA. Another switching arrangement was installed to disable transmissions going to the buffer system, as described previously. Figure 6 is an overall schematic of the hardware interface.

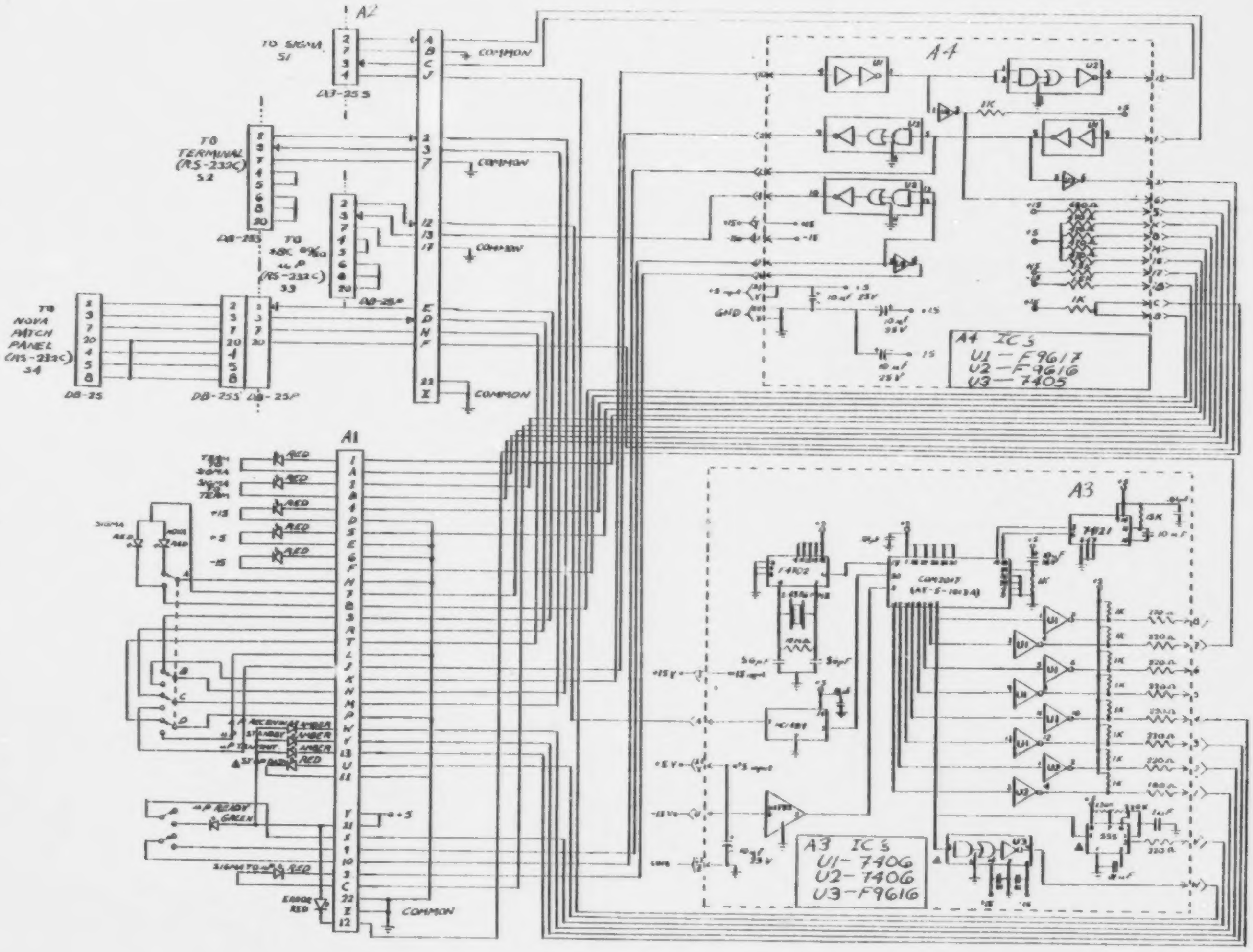

A2
TO SIGMA S1
DB-25S
COMMON
TO TERMINAL (RS-232C) S2
DB-25S
TO SBC 80/10 μP (RS-232C) S3
DB-25P
TO NOVA PATCH PANEL (RS-232C) S4
DB-25
DB-25S
DB-25P
A4
1K
GND
10 μf 25V
A4 ICS
U1 - F9617
U2 - F9616
U3 - 7405
A1
RED
AMBER
GREEN
ERROR RED
SIGMA TO μP RED
μP READY
COMMON
A3
COM2017 (AY-5-1013A)
555
7421
MC1488
A3 ICS
U1 - 7406
U2 - 7406
U3 - F9616

FIGURE 6. SCHEMATIC OF THE MONITOR INTERFACE.

SECTION 6

SOFTWARE IMPLEMENTATION

The software of the buffer system can be separated into two main divisions: the non-interrupt driven software and the interrupt driven (real-time) software. The real-time software has the responsibility of receiving all the information being transmitted to the buffer system. The reception of the information is the highest priority task in the operation of the buffer system.

6.1 NON-INTERRUPT DRIVEN SOFTWARE

The non-interrupt driven software as shown in Figure 7 is composed of four main modules: SETUP, WAIT, NOVAST, and OPER. The sequence of operations being accomplished in Figure 7 can be interrupted at any instant, and the sequence can be resumed without affecting the integrity of the system.

As can be seen from the figure, the module "SETUP" is only activated upon system initialization. SETUP contains the field programmable features of the buffer system. The operator must input the "software flagged" information the buffer system will use to activate the store-and-forward features of the buffer system.

The software flags allow the buffer system to discriminate between data which should be stored versus data which should be ignored. Five "starting characters" are used to activate the storage feature of the buffer system. Each of the three programmable ports which continuously monitor the intelligent instruments has individually programmed starting characters. Likewise, five "stopping" characters are input during setup for each of the individually programmed input ports.

Finally each of the three ports has a separate NOVA account into which the data will be transferred. When the MBS logs into the Data General multi-user BASIC system, the NOVA, asks for the account (specific disk directory) the user wishes to log into. This accounting information is distinct for each of the three ports. In the buffer system this information is stored as a sequence of four characters.

The last of the four programmable features to be input during the SETUP module is the baud rate for the specific ports. Each port (excluding the console port) is programmed for a specific baud rate during the SETUP module. Baud rate choices are limited to 110, 300 and 1200. The NOVA port is likewise programmed for a specific baud rate along with the three

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD; BEGIN SECTIONS HERE

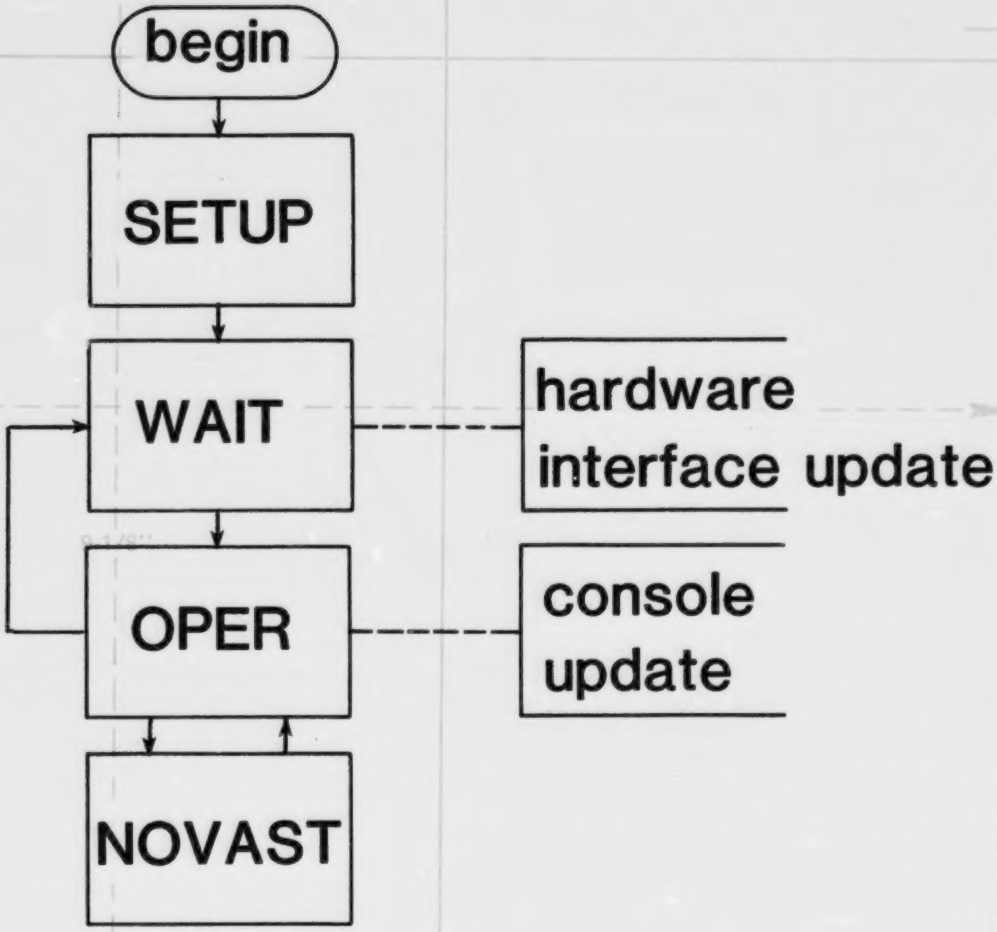

begin
SETUP
WAIT
hardware interface update
OPER
console update
NOVAST

FIGURE 7.

NON-INTERRUPT DRIVEN SOFTWARE.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8''
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

intelligent instrument ports. All of the field programmability of the buffer system is accomplished during the execution of the SETUP module. The SETUP module is responsible for the initialization of the programmable interrupt controllers, programmable timers (baud rate generators), and programmable UARTs.

During SETUP, the memory of the buffer system is polled to determine the memory size. The existing memory is then divided into 1000-byte blocks for use in the buffering operations. More detailed information on the SETUP module is contained in Section 7.1.1.

In Figure 7, the WAIT-OPER-NOVAST loop is the lowest priority operation of the system. After the SETUP module has been executed, the WAIT module updates the hardware interface status information approximately every 2 seconds (see Section 5.4). The WAIT module then calls the OPER module. The OPER module updates the diagostic console, which is displaying the real time status of the buffer system on a Hewlett-Packard 2640 series terminal. During the console update, the NOVAST module is called at approximately 120 times per second. For more detailed information on the OPER module see Section 7.4.

When the NOVAST module is called, it may perform one of several functions:

- Check to see if a report is pending for transmission to the NOVA;
- continue sending a report to the NOVA;
- continue logging onto the NOVA;
- continue logging off from the NOVA.

The NOVAST module is not interrupt driven. It mimics a human BASIC user on the NOVA. The NOVAST module will unmask the NOVA interrupt handler to enable the buffer system to receive in real-time a character from the NOVA whenever a character is due to be transmitted. By enabling the NOVA interrupt handler, the buffer system ensures the reception of any transmission from the NOVA.

The NOVAST module is programmed to respond to any foreseeable NOVA response. The NOVAST module works in close conjunction with the NOVA interrupt handler routine. The NOVAST module will send a transmission (ASCII character) to the NOVA, and the NOVAST module will activate the NOVA interrupt handler so that the echoed character can be received and checked against the transmitted character. If the characters do not match, a delete character is transmitted to the NOVA, and the correct character is again transmitted to the NOVA.

The MBS is programmed to recover from any type of transmission failure to the NOVA. If at any time during the transmission of a report to the NOVA, the NOVA fails to respond properly, the buffer system attempts to correct the malfunction; and if all else fails, the buffer system will re-initiate the login procedure and retransmit the report to the NOVA.

Even catastrophic failures such as the NOVA "being down" were accounted for. The buffer system simply attempts to long into the NOVA at two-minute intervals until the NOVA responds to the "log in" attempts.

The NOVAST module is in essence a 120-character-per-second typist that logs into the NOVA, types a buffered report into a disk file on the NOVA, and then logs off of the NOVA. The NOVAST module has been programmed to respond in a human fashion. More patience was programmed into the module, however; the timeout for a "no system response" condition is twenty minutes. For more detailed information on NOVAST see Section 7.6.

6.2 REAL-TIME SOFTWARE

The interrupt driven portion of the buffer system is shown in Figure 8. Module TB534 determines which UART has interrupted the microprocessor and directs the program flow to the appropriate vectored interrupt module. The modules USER0, USER1, and USER2 control the interpretation of the software flags which determine the store-and-forward features of the system. The three modules operate totally independently of each other. For example, USER0 can be attempting to identify the starting sequence characters while USER2 may be storing the third 1000-byte block of buffered information.

FRESH is the module which allocates the memory blocks to the appropriate users. Memory is dynamically allocated to the users. As a block is freed, it is immediately placed in the free block pool. If the FRESH module cannot locate a free block, the buffer system immediately halts further characters from being sent to it by dropping the "request to send" flags. In more advanced intelligent instrument systems, the dropping of this flag inhibits any further transmission of data from the intelligent instrument. FRESH also causes an alarm to be sent to a hardware interfaces at the intelligent instruments, thereby alerting the operators of the condition present. As the NOVAST module transmits buffered information to the NOVA, the memory blocks will be freed, and the fresh module will allow resumption of the data transfer to the buffer system.

The NOVA interrupt handler is responsible for receiving and appropriately handling the information being transmitted to the buffer system by the NOVA. The NOVA interrupt handler has multiple functions controlled by the NOINST (NOVA interrupt status) flag. The handler may process only one character. This is the procedure followed when the buffered data is being transferred to the NOVA and a character echo check is being imposed on the system. Or the handler may process a sequence of ASCII information being transmitted to the buffer system. This type of action is necessary when the NOVA responds to the buffer system logging into the NOVA. That is, when the buffer system sends an escape character, the NOVA may respond with the sequence "ACCOUNT ID:" or "ALL LINES BUSY." The NOVA interrupt handler will appropriately respond with the correct course of action.

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD; BEGIN SECTIONS HERE

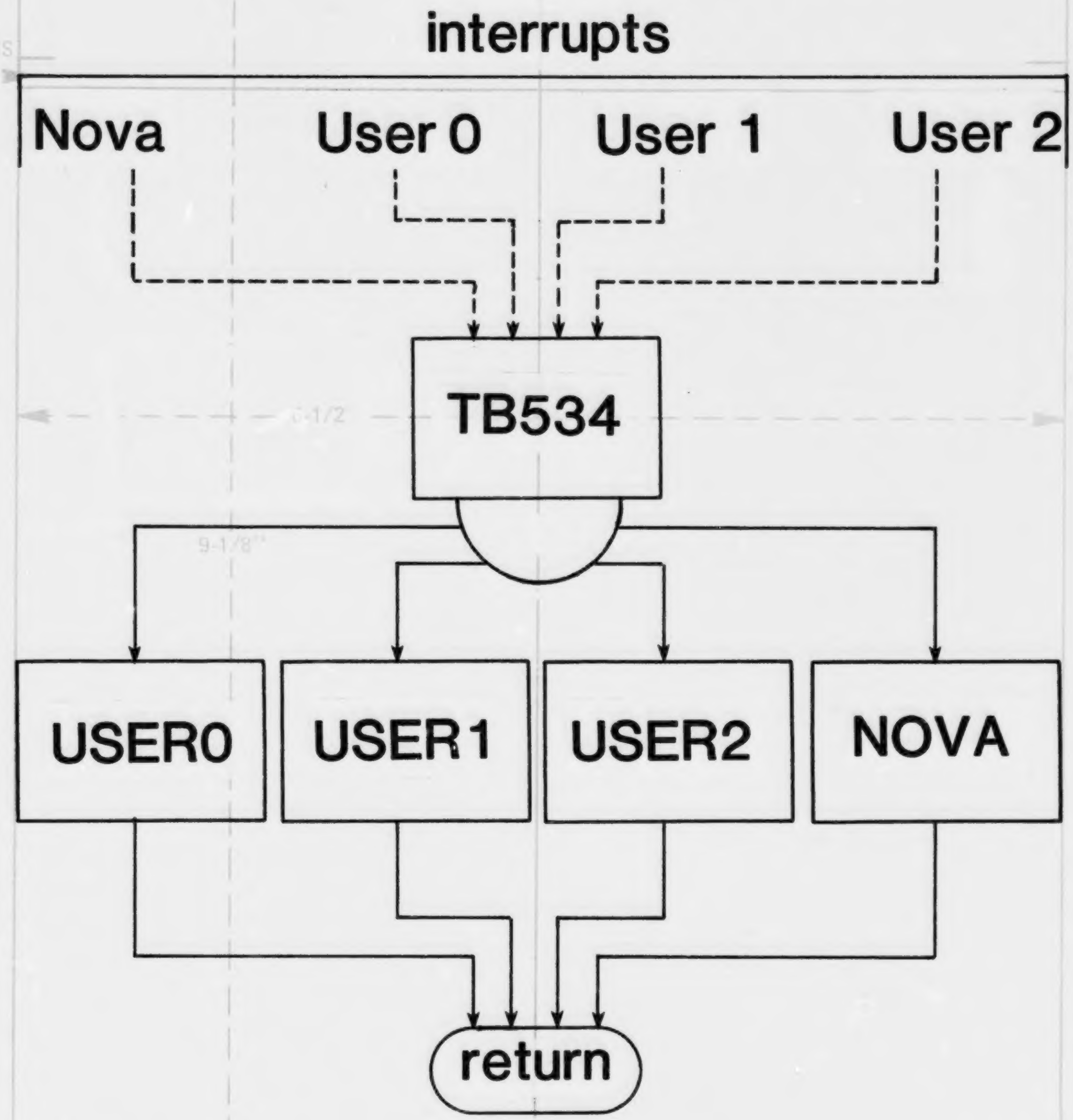

interrupts
Nova
User 0
User 1
User 2
TB534
USER0
USER1
USER2
NOVA
return

FIGURE 8.

INTERRUPT DRIVEN SOFTWARE.

6-1/2
9-1/8"
BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

The NOVA interrupt handler and the NOVAST module operate in close association with each other. For more detailed information, see Section 7.6. The combination of these two modules totally simulates a 120-character-per-second human typist.

SECTION 7

PROGRAM DOCUMENTATION

Section 6, software implementation, discussed the seven modules of code which make up the software of the MBS. In Section 6, the overall operation of the seven modules was discussed along with the interaction between specific modules. The program documentation, Section 7, describes in detail the operation of the specific modules of code. Each module of code is documented in sufficient detail so as to allow a competent software person the ability to modify, enhance, or otherwise alter the system to meet the specific needs of another facility. Each module of code has the following documentation associated with it:

- a verbal description of the module
- a corresponding flowchart
- the commented source listings
- a symbol table
- a symbol cross reference table.

This documentation should provide sufficient information to allow future support or enhancement of the MBS.

7.1 SETUP

SETUP is the first module of the Microprocessor Buffer System. As its name implies, the module initializes variables, inputs the operator-selectable parameters, and programs the iSBC 534 Four Channel Communications Expansion Board. SETUP is executed when the operator types the G800 command to the monitor during the booting procedure. When SETUP has been completed, control is passed to the WAIT module, and never returns to SETUP.

As SETUP begins, it first disables all interrupts. This must be done to protect the system from being interrupted before it is prepared to process interrupts. SETUP then initializes the stack pointer and prints the headings on the system console.

Next, SETUP initializes the iSBC 534 Board by performing a write to I/O address $1F_H$, and it selects the control block for that board by writing to

I/O address 1C$_H$. In both cases, the actual character written to these addresses is irrelevant.

The program then inputs the operator-selectable parameters for each of the four ports. The five starting characters, five stopping characters, and account ID for each of the three users are input and stored in TTYBUF. The baud rate for each of the four ports is input and used to program the programmable interval timer (PIT) for the port.

There are two programmable interval timers (PIT0 and PIT1) on the iSBC 534 Board. PIT0 contains Counter 0, Counter 1, and Counter 2 which are used to generate baud rate clock signals for ports 0, 1, and 2 respectively. PIT1 contains Counter 3, which is used to generate baud rate clock signals for port 3, and Counters 4 and 5, which are not currently used. There are two steps in initializing each counter. First, the system writes a PIT Mode Control Word for each counter to the appropriate I/O address. The PIT Mode Control Words used by this system set all of the first four counters so that they generate square waves for baud rate clock signals, and so that they use one-byte binary counters to count the number of 1.2288 MHz clock pulses per baud rate clock pulse. This is accomplished by writing a 16$_H$ to I/O address 13$_H$ for Counter 0, a 56$_H$ to I/O address 13$_H$ for Counter 1, a 96$_H$ to I/O address 13$_H$ for Counter 2, and a 16$_H$ to I/O address 17$_H$ for Counter 3. Secondly, the system writes the proper one-byte downcount number to each counter. The downcount number is AF$_H$ for 110 baud, 40$_H$ for 300 baud, or 10$_H$ for 1200 baud. The downcount number is written to I/O address 10$_H$ for Counter 0, I/O address 11$_H$ for Counter 1, I/O address 12$_H$ for Counter 2, and I/O address 14$_H$ for Counter 3.

Next, the Microprocessor Buffer System initializes the Universal Asynchronous/Synchronous Receiver/Transmitter (UART) for each port. To do this, it first writes to I/O address 1D$_H$ to select the data block of the iSBC 534 Board. The actual character written to that address is unimportant. Secondly, the system writes the Asynchronous Mode Instruction Word to each of the four UARTs. The Asynchronous Mode Instruction Word used by this system is 4F$_H$, which sets the UARTs for one stop bit, parity disabled, eight-bit character length, and a 64X baud rate factor. This word is written to I/O addresses 11$_H$, 13$_H$, 15$_H$, and 17$_H$ for UARTs 0, 1, 2, and 3, respectively. Thirdly, the system writes a UART Command Instruction Word to the same four I/O addresses. The Command Instruction Word for the first three UARTS is 27$_H$, which sets the UARTs for request to send, receive enable, data terminal ready, and transmit enable. The Command Instruction Word for UART 3 is 25$_H$, which is the same as 27$_H$ except that the data terminal ready bit is not set. This allows UART 3 to be connected to the NOVA patch panel without monopolizing a NOVA port.

The Programmable Interrupt Controllers (PICs) are initialized next. There are two PICs (PIC0 and PIC1) on the iSBC 534 Board. Because the PIC chips are much more sophisticated than necessary for the iSBC Board, there are four steps to programming them, but most of the steps are trivial. Step one is to write the first Initialization Command Word to I/O addresses 18$_H$ and 1A$_H$ for PIC0 and PIC1, respectively. The command word used is 16$_H$, although the only bit which is really critical is the one which informs the PICs that there are no slave PIC chips. Step two is to write the second Initialization Command Word to I/O addresses 19_H and $1B_H$ for PIC0 and PIC1, respectively. This command word is arbitrarily set at 00_H and serves no purpose other than to satisfy the PIC's need for a second Initialization Command Word. The third step is to write the first Operation Control Word to the PICs to mask unused interrupts. The first Operation Control Word for PIC0 is an AA_H written to I/O address 19_H; this masks interrupts 1, 3, 5, and 7. The first Operation Control Word for PIC1 is an FF_H written to I/O address $1B_H$; this masks all eight interrupts on PIC1. The final step of initializing the PICs is to set PIC0 to the auto-rotating priority mode by writing a second Operation Control Word to PIC0 at I/O address 18_H. This word is $A0_H$.

The programmable features of the iSBC 534 Board are discussed in more detail in the Intel Corporation publication iSBC 534 Four Channel Communication Expansion Board Hardware Reference Manual, Chapter 3, "Programming Information."

The SETUP module then clears the console screen and prints all of the titles and column headings there. SETUP copies the interrupt jump table from EPROM into RAM, beginning at RAM location $3FF0_H$. Finally, SETUP determines the size of the RAM available to the system. To test for 48K of RAM, SETUP attempts to change byte $FFFF_H$. If that fails, SETUP tests for 32K of RAM by attempting to change byte $BFFF_H$. If that also fails, SETUP assumes there is 16K of RAM present. SETUP sets every byte of memory to zero, initializes the system parameters, enables the interrupts, and passes control to the module named WAIT.

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
TOP OF IMAGE AREA
DROPPED HEAD BEGIN SECTIONS HERE
9-1/8"

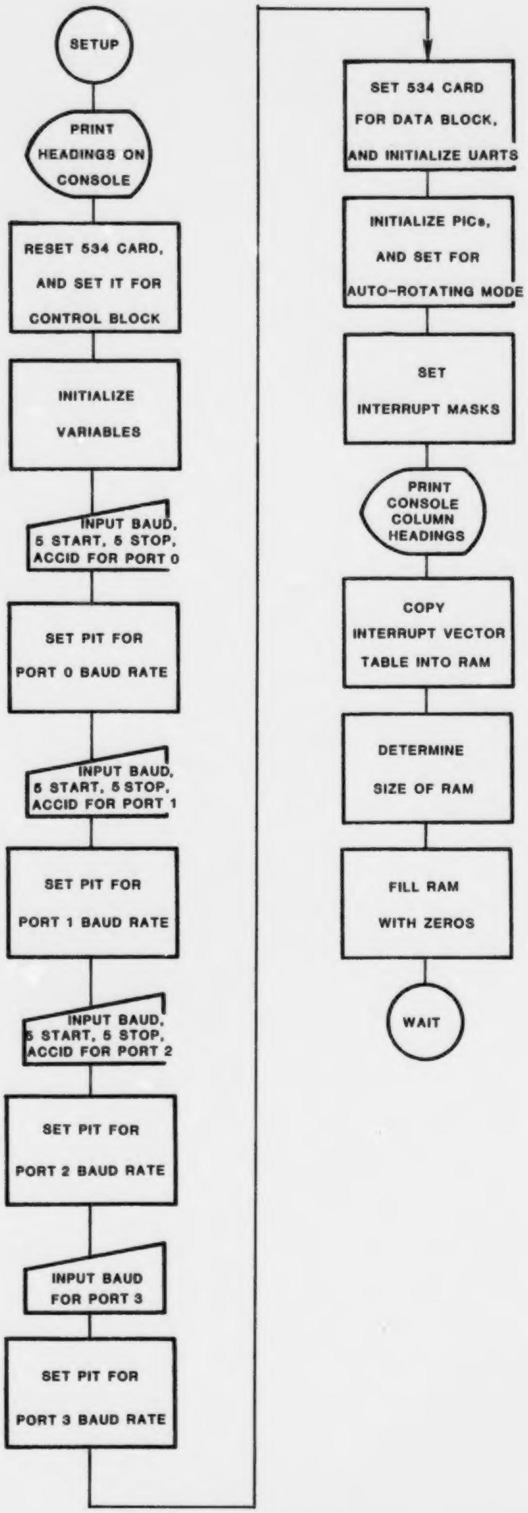

SETUP
PRINT HEADINGS ON CONSOLE
RESET 534 CARD, AND SET IT FOR CONTROL BLOCK
INITIALIZE VARIABLES
INPUT BAUD, 5 START, 5 STOP, ACCID FOR PORT 0
SET PIT FOR PORT 0 BAUD RATE
INPUT BAUD, 5 START, 5 STOP, ACCID FOR PORT 1
SET PIT FOR PORT 1 BAUD RATE
INPUT BAUD, 5 START, 5 STOP, ACCID FOR PORT 2
SET PIT FOR PORT 2 BAUD RATE
INPUT BAUD FOR PORT 3
SET PIT FOR PORT 3 BAUD RATE
SET 534 CARD FOR DATA BLOCK, AND INITIALIZE UARTS
INITIALIZE PICs, AND SET FOR AUTO-ROTATING MODE
SET INTERRUPT MASKS
PRINT CONSOLE COLUMN HEADINGS
COPY INTERRUPT VECTOR TABLE INTO RAM
DETERMINE SIZE OF RAM
FILL RAM WITH ZEROS
WAIT

FIGURE 9. SETUP FLOWCHART.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA. OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"

EPA-287 (Cin.)
(4-76)

```
ASM80 :F1:SETUP.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    1

  LOC  OBJ         LINE        SOURCE STATEMENT

                      1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      2 ;
                      3 ;         MICROCOMPUTER DATA BUFFER SYSTEM
                      4 ;
                      5 ;         BY:  R. A. JOLLIS (EMSL)
                      6 ;              J. M. TEUSCHLER (EMSL)
                      7 ;              J. E. KOPKE (UCCC)
                      8 ;
                      9 ;         MODULE NAMES (IN ORDER):  SETUP
                     10 ;                                   TB534
                     11 ;                                   WAIT
                     12 ;                                   OPER
                     13 ;                                   USER
                     14 ;                                   NOVAST
                     15 ;                                   NOVA
                     16 ;
                     17 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     18 ;
                     19 ;
                     20 ;
                     21 ;
                     22 ;
                     23 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     24 ;
                     25 ;         MODULE NAME:  SETUP
                     26 ;
                     27 ;         DESCRIPTION:  THIS MODULE SETS UP THE SBC534 COMMUNICATIONS
                     28 ;                       CARD AND INITIALIZES MEMORY.  IT SETS UP
                     29 ;                       INTERRUPT VECTORS AND SETS UP USER START AND
                     30 ;                       STOP STRINGS.  THIS MODULE ALSO CONTAINS
                     31 ;                       SEVERAL SUBROUTINES THAT ARE USED THROUGHOUT
                     32 ;                       THE SYSTEM.
                     33 ;
                     34 ;         MOST RECENT REVISION:  AUGUST 22, 1980
                     35 ;
                     36 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     37 ;
                     38         NAME    SETUP
                     39         PUBLIC  CO,CI,UARTLI,UARTLO,PIT5,MSG,TTYBUF,BOTTOM,TOP,BASE,STAT,SYSTAT
                     40         PUBLIC  BLK0,BLK1,BLK2,SEQ0,SEQ1,SEQ2,STBY0,STBY1,STBY2,OVRFLW,ZERO1
                     41         PUBLIC  SENTO,CHAAMT,CHAPTR,LSTUSR,NOINST,NOVSTS,VECFLG
                     42         PUBLIC  ROUPTR,RUNFLG,SNTP1,TIMCTR,TIMCT1,USRFLG,CURBLK,FSTBLK
                     43         EXTRN   TB534,WAIT,EOI,HOME,TAB3,UPRDY,UPRX,UPSTBY;TIMOUT
  0000               44 SBC     SET     00H             ;CONDITIONAL ASSEMBLY FLAG (SBC-00 MDS-FF)
                     45         IF SBC EQ 00H
  0307               46 CO      EQU     0307H           ;CONSOLE OUTPUT ROUTINE IN 80/20 MONITOR
  02F4               47 CI      EQU     02F4H           ;CONSOLE INPUT ROUTINE IN 80/20 MONITOR
  3C00               48 BASE    EQU     3C00H           ;BASE OF UTILITY STORAGE IN SBC-80/20-4
  3FF0               49 INT4    EQU     3FF0H           ;INTERRUPT JUMP TABLE IN SBC-80/20-4
                     50         ELSE
                     51 CO      EQU     0FD1DH          ;CONSOLE OUTPUT ROUTINE IN MDS-800 MONITOR
                     52 CI      EQU     0FC85H          ;CONSOLE INPUT ROUTINE IN MDS-800 MONITOR
                     53 BASE    EQU     0EC00H          ;BASE OF UTILITY STORAGE IN MDS-800
                     54 INT4    EQU     0020H           ;INTERRUPT JUMP TABLE IN MDS-800
                     55         ENDIF
  004F               56 UARTMO  EQU     4FH             ;UART DATA FORMAT
  0027               57 UARTLI  EQU     27H             ;UART:RTS,RX,DTR,TX
  0025               58 UARTLO  EQU     25H             ;UART:RTS,RX,TX
```

EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE
DROPPED HEAD, BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    2

 LOC  OBJ         LINE        SOURCE STATEMENT

 4000              59 BOTTOM  EQU     4000H          ;BOTTOM OF MEMORY
 3C10              60 STAT    EQU     BASE+10H       ;USER STATUS
 3C12              61 BLK0    EQU     BASE+12H       ;BLOCK COUNTER
 3D12              62 BLK1    EQU     BLK0+100H      ;BLOCK COUNTER
 3E12              63 BLK2    EQU     BLK0+200H      ;BLOCK COUNTER
 3C11              64 SEQ0    EQU     BASE+11H       ;SEQUENCE COUNTER
 3D11              65 SEQ1    EQU     SEQ0+100H      ;SEQUENCE COUNTER
 3E11              66 SEQ2    EQU     SEQ0+200H      ;SEQUENCE COUNTER
 3C15              67 STBY0   EQU     BASE+15H       ;REPORT COUNTER
 3D15              68 STBY1   EQU     STBY0+100H     ;REPORT COUNTER
 3E15              69 STBY2   EQU     STBY0+200H     ;REPORT COUNTER
 3F10              70 SYSTAT  EQU     STAT+300H      ;SYSTEM STATUS
 3F20              71 TOP     EQU     SYSTAT+10H     ;TOP OF MEMORY STORAGE
 3F11              72 OVRFLW  EQU     SYSTAT+01H     ;OVERFLOW STATUS
 3CB0              73 TTYBUF  EQU     BASE+0B0H      ;START/STOP/DIR STORAGE BASE ADDRESS
 0016              74 PIT03   EQU     16H            ;UART COUNTERS 0 AND 3 FOR BAUD RATE
 0056              75 PIT14   EQU     56H            ;UART COUNTERS 1 AND 4 FOR BAUD RATE
 0096              76 PIT25   EQU     96H            ;UART COUNTERS 2 AND 5 FOR BAUD RATE
 0076              77 PIT4    EQU     76H            ;UART COUNTER 5 FOR TIMER (ASYNC)
 00B0              78 PIT5    EQU     0B0H           ;UART COUNTER 5 FOR TIMER
 3C18              79 SENT0   EQU     BASE+18H       ;NUMBER OF REPORTS SENT FOR EACH USER
 3F13              80 CHAPTR  EQU     SYSTAT+3H      ;CHARACTER POINTER
 3F15              81 LSTUSR  EQU     CHAPTR+2H      ;LAST USER
 3F16              82 NOINST  EQU     LSTUSR+1H      ;NOVA INTERRUPT STATUS
 3F17              83 NOVSTS  EQU     NOINST+1H      ;NOVA STATUS
 3F18              84 ROUPTR  EQU     NOVSTS+1H      ;ROUTINE POINTER
 3F1A              85 SNTP1   EQU     ROUPTR+2H      ;SENTn + 1
 3F1B              86 TIMCTR  EQU     SNTP1+1H       ;TIME COUNTER FOR NUMBER OF MINUTES
 3F1C              87 TIMCT1  EQU     TIMCTR+1H      ;TIME COUNTER FOR NUMBER OF CALLS TO NOVAST
 3F1E              88 USRFLG  EQU     TIMCT1+2H      ;USER FLAG
 3F1F              89 CURBLK  EQU     USRFLG+1H      ;CURRENT BLOCK MSB
 3F30              90 FSTBLK  EQU     SYSTAT+20H     ;FIRST BLOCK MSG
 3F31              91 CHAAMT  EQU     FSTBLK+1H      ;CHARACTER AMOUNT
 3F33              92 RUNFLG  EQU     CHAAMT+2H      ;FLAG INDICATING THAT CWECHO IS SENDING RUN"MICROIN
                      PUT
 3F34              93 VECFLG  EQU     RUNFLG+1H      ;FLAG INDICATING LIGHT OR DARK VECTOR MODE
                   94         STKLN   200H           ;SET STACKLENGTH IN MEMORY ROUTINE IN MONITOR
                   95         DSEG                   ;START DATA SEGMENT
                   96 ;
                   97 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   98 ;
                   99 ;     ROUTINE NAME:  SETUP
                  100 ;
                  101 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                  102 ;
                  103 SETUP:  IF SBC EQ 0
 0000 F3          104         DI                     ;DISABLE INTERRUPTS
                  105         ELSE
                  106         EI                     ;ENABLE INTERRUPTS
                  107         ENDIF
 0001 310000   S  108         LXI     SP,STACK       ;INITIALIZE STACK POINTER
 0004 01E902   D  109         LXI     B,TITLE        ;  PRINT HEADER
 0007 CDD001   D  110         CALL    MSG            ;  TO CONSOLE
 000A 01DB03   D  111         LXI     B,MEMLK        ;    LOCK MEMORY
 000D CDD001   D  112         CALL    MSG            ;    FOR INITIALIZATION
 0010 D31F        113         OUT     1FH            ;RESET 534 UART CARD
 0012 D31C        114         OUT     1CH            ;SET CONTROL BLOCK
                  115 ;
                  116 ;
```

BEGIN LAST LINE OF TEXT
3/8''
DIMENSION FOR TABLES AND ILLUSTRATIONS

PAGE NUMBER
EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    3

 LOC  OBJ          LINE        SOURCE STATEMENT

                    117 ;       *****INITIALIZE NOVA VARIABLES*****
                    118 ;
                    119 ;
 0014 21133F        120         LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
 0017 3E00          121         MVI     A,00H           ;INITIALIZE COUNTER IN A-REG AT ZERO
 0019 3600          122 NXTBYT: MVI     M,00H           ;SET BYTE TO ZERO
 001B 23            123         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
 001C 3C            124         INR     A               ;INCREMENT COUNTER IN A-REG
 001D FE0D          125         CPI     0DH             ;IS COUNTER UP TO 13?
 001F C21900   D    126         JNZ     NXTBYT          ;IF NO, JUMP BACK TO ZERO NEXT BYTE
 0022 21303F        127         LXI     H,FSTBLK        ;LOAD ADDRESS OF FSTBLK INTO H-L-REG
 0025 3600          128         MVI     M,00H           ;SET FSTBLK TO ZERO
 0027 23            129         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-MSB
 0028 3600          130         MVI     M,00H           ;SET CHAAMT-MSB TO ZERO
 002A 23            131         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
 002B 3600          132         MVI     M,00H           ;SET CHAAMT-LSB TO ZERO
 002D 23            133         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF RUNFLG
 002E 3600          134         MVI     M,00H           ;SET RUNFLG TO ZERO
 0030 23            135         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF VECFLG
 0031 3600          136         MVI     M,00H           ;SET VECFLG TO ZERO
 0033 21183C        137         LXI     H,SENT0         ;LOAD ADDRESS OF SENT0 INTO H-L-REG
 0036 3600          138         MVI     M,00H           ;SET SENT0 TO ZERO
 0038 24            139         INR     H               ;INCREMENT H-REG TO FORM ADDRESS OF SENT1
 0039 3600          140         MVI     M,00H           ;SET SENT1 TO ZERO
 003B 24            141         INR     H               ;INCREMENT H-REG TO FORM ADDRESS OF SENT2
 003C 3600          142         MVI     M,00H           ;SET SENT2 TO ZERO
 003E 211E3F        143         LXI     H,USRFLG        ;LOAD ADDRESS OF USRFLG IN TO H-L-REG
 0041 36FF          144         MVI     M,0FFH          ;SET USRFLG TO FF
                    145 ;
                    146 ;
                    147 ;       *****INPUT PARAMETERS FOR PORT 0*****
                    148 ;
                    149 ;
 0043 3E16          150         MVI     A,PIT03         ;ENABLE COUNTERS
 0045 D313          151         OUT     13H             ;COUNTER 0
 0047 D317          152         OUT     17H             ;COUNTER 3
 0049 01E103   D    153         LXI     B,PRMT0         ;  PRINT PROMPT
 004C CDD001   D    154         CALL    MSG             ;  FOR PORT 0
 004F CDC001   D    155         CALL    INCO            ;READ FROM CONSOLE
 0052 CDE601   D    156         CALL    BAUD            ;CHECK BAUD RATE
 0055 D310          157         OUT     10H             ;SET SPEED PORT 0
 0057 017604   D    158         LXI     B,STRTCH        ;  PRINT START
 005A CDD001   D    159         CALL    MSG             ;  CHARACTER PROMPT
 005D 21B03C        160         LXI     H,TTYBUF        ;LOAD STORE ADDRESS
 0060 1E05          161         MVI     E,05H           ;5 CHARACTERS
 0062 CD1802   D    162         CALL    STCH            ;STORE CHARACTERS
 0065 019F04   D    163         LXI     B,STOPCH        ;  PRINT STOP
 0068 CDD001   D    164         CALL    MSG             ;  CHARACTER PROMPT
 006B 21B53C        165         LXI     H,TTYBUF+05H    ;LOAD STORE ADDRESS
 006E 1E05          166         MVI     E,05H           ;5 CHARACTERS
 0070 CD1802   D    167         CALL    STCH            ;STORE CHARACTERS
 0073 01C804   D    168         LXI     B,ACCTID        ;  PRINT ACCOUNT
 0076 CDD001   D    169         CALL    MSG             ;  ID PROMPT
 0079 21BA3C        170         LXI     H,TTYBUF+0AH    ;LOAD STORE ADDRESS
 007C 1E04          171         MVI     E,04H           ;4 CHARACTERS
 007E CD1802   D    172         CALL    STCH            ;STORE CHARACTERS
 0081 23            173         INX     H               ;INCREMENT H-L-REG TO ADDRESS AFTER ID
 0082 360D          174         MVI     M,0DH           ;PUT CARRIAGE RETURN AFTER ID
 0084 01E304   D    175         LXI     B,SETMSG        ;  PRINT PORT SET
```

BEGIN LAST LINE OF TEXT
3/8"
M OF AREA, DE DIMENSION FOR TABLES AND ILLUSTRATIONS
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE
DROPPED HEAD; BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    4

  LOC  OBJ         LINE        SOURCE STATEMENT

  0087 CDD001   D   176         CALL    MSG             ;  MESSAGE
                    177 ;
                    178 ;
                    179 ;       *****INPUT PARAMETERS FOR PORT 1*****
                    180 ;
                    181 ;
  008A 3E56         182         MVI     A,PIT14         ;ENABLE COUNTER
  008C D313         183         OUT     13H             ;COUNTER 1
  008E 011505   D   184         LXI     B,PRMT1         ;  PRINT PROMPT
  0091 CDD001   D   185         CALL    MSG             ;  FOR PORT 1
  0094 015604   D   186         LXI     B,ENTSP         ;    PRINT ENTER
  0097 CDD001   D   187         CALL    MSG             ;    SPEED PROMPT
  009A CDC001   D   188         CALL    INCO            ;READ FROM CONSOLE
  009D CDE601   D   189         CALL    BAUD            ;CHECK BAUD RATE
  00A0 D311         190         OUT     11H             ;SET SPEED PORT 1
  00A2 017604   D   191         LXI     B,STRTCH        ;  PRINT START
  00A5 CDD001   D   192         CALL    MSG             ;  CHARACTER PROMPT
  00A8 21B03D       193         LXI     H,TTYBUF+100H   ;LOAD STORE ADDRESS
  00AB 1E05         194         MVI     E,05H           ;5 CHARACTERS
  00AD CD1802   D   195         CALL    STCH            ;STORE CHARACTERS
  00B0 019F04   D   196         LXI     B,STOPCH        ;  PRINT STOP
  00B3 CDD001   D   197         CALL    MSG             ;  CHARACTER PROMPT
  00B6 21B53D       198         LXI     H,TTYBUF+105H   ;LOAD STORE ADDRESS
  00B9 1E05         199         MVI     E,05H           ;5 CHARACTERS
  00BB CD1802   D   200         CALL    STCH            ;STORE CHARACTERS
  00BE 01C804   D   201         LXI     B,ACCTID        ;  PRINT ACCOUNT
  00C1 CDD001   D   202         CALL    MSG             ;  ID PROMPT
  00C4 21BA3D       203         LXI     H,TTYBUF+10AH   ;LOAD STORE ADDRESS
  00C7 1E04         204         MVI     E,04H           ;4 CHARACTERS
  00C9 CD1802   D   205         CALL    STCH            ;STORE CHARACTERS
  00CC 23           206         INX     H               ;INCREMENT H-L-REG TO ADDRESS AFTER ID
  00CD 360D         207         MVI     M,0DH           ;PUT CARRIAGE RETURN AFTER ID
  00CF 01E304   D   208         LXI     B,SETMSG        ;  PRINT PORT
  00D2 CDD001   D   209         CALL    MSG             ;  SET MESSAGE
                    210 ;
                    211 ;
                    212 ;       *****INPUT PARAMETERS FOR PORT 2*****
                    213 ;
                    214 ;
  00D5 3E96         215         MVI     A,PIT25         ;ENABLE COUNTER
  00D7 D313         216         OUT     13H             ;COUNTER 2
  00D9 012505   D   217         LXI     B,PRMT2         ;  PRINT PROMPT
  00DC CDD001   D   218         CALL    MSG             ;  FOR PORT 2
  00DF 015604   D   219         LXI     B,ENTSP         ;    PRINT ENTER
  00E2 CDD001   D   220         CALL    MSG             ;    SPEED PROMPT
  00E5 CDC001   D   221         CALL    INCO            ;READ FROM CONSOLE
  00E8 CDE601   D   222         CALL    BAUD            ;CHECK BAUD RATE
  00EB D312         223         OUT     12H             ;SET SPEED PORT 2
  00ED 017604   D   224         LXI     B,STRTCH        ;  PRINT START
  00F0 CDD001   D   225         CALL    MSG             ;  CHARACTER PROMPT
  00F3 21B03E       226         LXI     H,TTYBUF+200H   ;LOAD STORE ADDRESS
  00F6 1E05         227         MVI     E,05H           ;5 CHARACTERS
  00F8 CD1802   D   228         CALL    STCH            ;STORE CHARACTERS
  00FB 019F04   D   229         LXI     B,STOPCH        ;  PRINT STOP
  00FE CDD001   D   230         CALL    MSG             ;  CHARACTER PROMPT
  0101 21B53E       231         LXI     H,TTYBUF+205H   ;LOAD STORE ADDRESS
  0104 1E05         232         MVI     E,05H           ;5 CHARACTERS
  0106 CD1802   D   233         CALL    STCH            ;STORE CHARACTERS
  0109 01C804   D   234         LXI     B,ACCTID        ;  PRINT ACCOUNT
```

BEGIN LAST LINE OF TEXT
3/8"
DIMENSION FOR TABLES AND ILLUSTRATIONS

PAGE NUMBER
EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD BEGIN SECTIONS HERE

```
  LOC  OBJ        LINE        SOURCE STATEMENT

  010C CDD001   D  235         CALL    MSG             ;  ID PROMPT
  010F 21BA3E      236         LXI     H,TTYBUF+20AH   ;LOAD STORE ADDRESS
  0112 1E04        237         MVI     E,04H           ;4 CHARACTERS
  0114 CD1802   D  238         CALL    STCH            ;STORE CHARACTERS
  0117 23          239         INX     H               ;INCREMENT H-L-REG TO ADDRESS AFTER ID
  0118 360D        240         MVI     M,0DH           ;PUT CARRIAGE RETURN AFTER ID
  011A 01E304   D  241         LXI     B,SETMSG        ;  PRINT PORT
  011D CDD001   D  242         CALL    MSG             ;  SET MESSAGE
                   243 ;
                   244 ;
                   245 ;       *****INPUT PARAMETERS FOR PORT 3*****
                   246 ;
                   247 ;
  0120 013505   D  248         LXI     B,PRMT3         ;    PRINT PROMPT
  0123 CDD001   D  249         CALL    MSG             ;    FOR PORT 3
  0126 015604   D  250         LXI     B,ENTSP         ;    PRINT ENTER
  0129 CDD001   D  251         CALL    MSG             ;    SPEED PROMPT
  012C CDC001   D  252         CALL    INCO            ;READ FROM CONSOLE
  012F CDE601   D  253         CALL    BAUD            ;CHECK BAUD RATE
  0132 D314        254         OUT     14H             ;SET SPEED PORT 3
  0134 01E304   D  255         LXI     B,SETMSG        ;  PRINT PORT SET
  0137 CDD001   D  256         CALL    MSG             ;  MESSAGE TO CONSOLE
                   257 ;
                   258 ;
                   259 ;       *****INITIALIZE INTERRUPT TIMER*****
                   260 ;
                   261 ;
  013A 3E76        262         MVI     A,PIT4          ;ENABLE COUNTERS
  013C D317        263         OUT     17H             ;COUNTER 4
  013E AF          264         XRA     A               ;ZERO OUT ACCUMULATER
  013F D315        265         OUT     15H             ;  PRESET
  0141 3EF0        266         MVI     A,0F0H          ;  COUNTER 4
  0143 D315        267         OUT     15H             ;  FOR 50 mS
  0145 D31D        268         OUT     1DH             ;SET DATA BLOCK
                   269 ;
                   270 ;
                   271 ;       *****INITIALIZE USARTS*****
                   272 ;
                   273 ;
  0147 3E4F        274         MVI     A,UARTMO        ;SET DATA FORMAT
  0149 D311        275         OUT     11H             ;PORT 0
  014B D313        276         OUT     13H             ;PORT 1
  014D D315        277         OUT     15H             ;PORT 2
  014F D317        278         OUT     17H             ;PORT 3
  0151 3E27        279         MVI     A,UARTLI        ;ENABLE SIGNAL
  0153 D311        280         OUT     11H             ;PORT 0
  0155 D313        281         OUT     13H             ;PORT 1
  0157 D315        282         OUT     15H             ;PORT 2
  0159 3E25        283         MVI     A,UARTLO        ;ENABLE SIGNAL DTR OFF
  015B D317        284         OUT     17H             ;PORT 3
                   285 ;
                   286 ;
                   287 ;       *****INITIALIZE INTERRUPT CONTROLLERS*****
                   288 ;
                   289 ;
  015D 3E16        290         MVI     A,16H           ;INITIALIZE PICS
  015F D318        291         OUT     18H             ;PIC 0
  0161 D31A        292         OUT     1AH             ;PIC 1
  0163 AF          293         XRA     A               ;ZERO OUT ACCUMULATER
```

BEGIN LAST LINE OF TEXT
3/8"
PAGE NUMBER
DIMENSION FOR TABLES AND ILLUSTRATIONS
EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD: BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    6

 LOC  OBJ        LINE        SOURCE STATEMENT

 0164 D319        294         OUT     19H            ;PIC 0
 0166 D31B        295         OUT     1BH            ;PIC 1
 0168 3EA0        296         MVI     A,0A0H         ;  SET PIC TO
 016A D318        297         OUT     18H            ;  AUTO ROTATE
 016C 3EFF        298         MVI     A,0FFH         ;MASK INTERRUPTS
 016E D31B        299         OUT     1BH            ;0,1,2,3,4,5,6,7 PIC 1
 0170 3EAA        300         MVI     A,0AAH         ;MASK INTERRUPTS
 0172 D319        301         OUT     19H            ;1,3,5,7 PIC 0
                  302         IF SBC EQ 0FFH
                  303         MVI     A,12H          ;ICW1
                  304         OUT     0FDH           ;OUT TO PIC
                  305         XRA     A              ;ZERO ACCUMULATER
                  306         OUT     0FCH           ;OUT TO PIC
                  307         MVI     A,20H          ;SET MASKS (INT5)
                  308         OUT     0FCH           ;NO MASKS
                  309         ENDIF
                  310 ;
                  311 ;
                  312 ;       *****INITIALIZE CONSOLE *****
                  313 ;
                  314 ;
 0174 01DE03   D  315         LXI     B,MEMULK       ;      UNLOCK
 0177 CDD001   D  316         CALL    MSG            ;      MEMORY
 017A 01E902   D  317         LXI     B,TITLE        ;    OUTPUT TITLE
 017D CDD001   D  318         CALL    MSG            ;    TO CONSOLE
 0180 01B905   D  319         LXI     B,HEADER       ;  OUTPUT HEADER
 0183 CDD001   D  320         CALL    MSG            ;  TO CONSOLE
 0186 010C06   D  321         LXI     B,TABDAT       ;    OUTPUT TAB DATA
 0189 CDD001   D  322         CALL    MSG            ;    TO CONSOLE
 018C 21AB01   D  323         LXI     H,JPTB         ;LOAD JUMP TABLE ADDRESS
 018F 11F03F      324         LXI     D,INT4         ;LOAD JUMP TABLE
                  325         IF SBC EQ 0FFH
                  326         MVI     B,0FH          ;LOAD TABLE SIZE (MDS-800)
                  327         ELSE
 0192 0608        328         MVI     B,08H          ;LOAD TABLE SIZE (SBC-80/20-4)
                  329         ENDIF
                  330 ;
                  331 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                  332 ;
                  333 ;     ROUTINE NAME: LOAD
                  334 ;     DESCRIPTION:  THIS ROUTINE LOADS THE JUMP TABLE AT THE INTERRUPT
                  335 ;                   VECTOR LOCATIONS.
                  336 ;
                  337 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                  338 ;
 0194 7E          339 LOAD:   MOV     A,M            ;LOAD ACCUM WITH INSTRUCTION
 0195 12          340         STAX    D              ;STORE IN MEMORY
 0196 13          341         INX     D              ;INCREMENT MEMEORY
 0197 23          342         INX     H              ;INCREMENT TABLE
 0198 05          343         DCR     B              ;DECREMENT COUNTER
 0199 C29401   D  344         JNZ     LOAD           ;JUMP
 019C DB10        345         IN      10H            ;RESET RXRDY PORT 0
 019E DB12        346         IN      12H            ;RESET RXRDY PORT 1
 01A0 DB14        347         IN      14H            ;RESET RXRDY PORT 2
 01A2 DB16        348         IN      16H            ;RESET RXRDY PORT 3
 01A4 CD2602   D  349         CALL    MEMCHK         ;CALL MEMORY CHECK AND ZERO ROUTINE
 01A7 FB          350         EI                     ;ENABLE INTERRUPTS
 01A8 C30000   E  351         JMP     WAIT           ;JUMP TO WAIT LOOP
                  352 ;
```

BEGIN LAST LINE OF TEXT
DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"

PAGE NUMBER
EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD, BEGIN SECTIONS HERE

```
  LOC  OBJ         LINE        SOURCE STATEMENT

                    353 ;
                    354 ;       *****INTERRUPT JUMP TABLE*****
                    355 ;
                    356 ;
  01AB F3           357 JPTB:   DI                      ;DISABLE INTERRUPTS
  01AC C30000    E  358         JMP     TB534           ;JUMP TO 534 TABLE
                    359         IF SBC EQ OFFH
                    360         NOP                     ;FILLER (MDS-800)
                    361         NOP                     ;FILLER (MDS-800)
                    362         NOP                     ;FILLER (MDS-800)
                    363         NOP                     ;FILLER (MDS-800)
                    364         ENDIF
  01AF F3           365         DI                      ;DISABLE INTERRUPTS
  01B0 C3B301    D  366         JMP     TIMOUT          ;JUMP TO TIMEOUT ROUTINE
                    367         IF SBC EQ OFFH
                    368         NOP                     ;FILLER (MDS-800)
                    369         NOP                     ;FILLER (MDS-800)
                    370         NOP                     ;FILLER (MDS-800)
                    371         NOP                     ;FILLER (MDS-800)
                    372         ENDIF
                    373 ;
                    374 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    375 ;
                    376 ;     ROUTINE NAME: TIMOUT
                    377 ;     DESCRIPTION: THIS ROUTINE IS A TEMPORY RESPONSE TO AN INTERRUPT
                    378 ;                  TIMER INTERRUPT.  IT WOULD BE REPLACED BY A ROUTINE
                    379 ;                  THAT WOULD SERVICE THE INTERRUPT TIMER.
                    380 ;
                    381 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    382 ;
  01B3 F5           383 TIMOUT: PUSH    PSW             ;SAVE REGISTER
  01B4 C5           384         PUSH    B               ;SAVE REGISTER
  01B5 D5           385         PUSH    D               ;SAVE REGISTER
  01B6 E5           386         PUSH    H               ;SAVE REGISTER
  01B7 015E06    D  387         LXI     B,TIMER         ;LOAD MESSAGE
  01BA CDD001    D  388         CALL    MSG             ;OUTPUT TO CONSOLE
  01BD C30300    E  389         JMP     EOI+03H         ;END OF INTERRUPT
                    390 ;
                    391 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    392 ;
                    393 ;     ROUTINE NAME: INCO
                    394 ;     INPUTS: NONE
                    395 ;     OUTPUTS: C REG
                    396 ;     CALLS: GETCH,CO
                    397 ;     DESCRIPTION: THIS ROUTINE GETS A CHARACTER AND ECHOS IT ON THE
                    398 ;                  CONSOLE SCREEN.
                    399 ;
                    400 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    401 ;
  01C0 D5           402 INCO:   PUSH    D               ;SAVE REGISTER
  01C1 E5           403         PUSH    H               ;SAVE REGISTER
  01C2 F5           404         PUSH    PSW             ;SAVE REGISTER
  01C3 CDF402       405         CALL    CI              ;CALL CONSOLE INPUT ROUTINE
  01C6 E67F         406         ANI     7FH             ;TURN OFF PARITY BIT IF ITS ON
  01C8 4F           407         MOV     C,A             ;PUT VALUE IN C REG
  01C9 CD0703       408         CALL    CO              ;OUTPUT TO CONSOLE
  01CC F1           409         POP     PSW             ;RESET REGISTER
  01CD E1           410         POP     H               ;RESET REGISTER
  01CE D1           411         POP     D               ;RESET REGISTER
```

BEGIN LAST LINE OF TEXT
3/8"
DIMENSION FOR TABLES AND ILLUSTRATIONS
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    8

 LOC  OBJ          LINE        SOURCE STATEMENT

 01CF C9            412         RET                         ;RETURN WITH CHARACTER IN C REGISTER
                    413 ;
                    414 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    415 ;
                    416 ;      ROUTINE NAME: MSG
                    417 ;      INPUTS: ADDRESS IN BC PAIR
                    418 ;      OUTPUTS: NONE
                    419 ;      CALLS: CO
                    420 ;      DESCRIPTION: THIS ROUTINE OUTPUTS CHARACTERS FROM A GIVEN CHARACTER
                    421 ;                   STRING UNTIL IT ENCOUNTERS A NUL CHARACTER (00H) AT
                    422 ;                   WHICH TIME IT RETURNS TO THE POINT IT WAS CALLED.  BEFORE
                    423 ;                   OUTPUTING THE CHARACTER IT SETS THE PARITY BIT.
                    424 ;
                    425 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    426 ;
 01D0 0A            427 MSG:    LDAX    B                   ;GET LETTER
 01D1 C5            428         PUSH    B                   ;SAVE REGISTER
 01D2 D5            429         PUSH    D                   ;SAVE REGISTER
 01D3 E5            430         PUSH    H                   ;SAVE REGISTER
 01D4 F5            431         PUSH    PSW                 ;SAVE REGISTER
 01D5 F680          432         ORI     80H                 ;SET PARITY
 01D7 4F            433         MOV     C,A                 ;  SEND CHARACTER OUT
 01D8 CD0703        434         CALL    CO                  ;  TO CONSOLE
 01DB F1            435         POP     PSW                 ;RESET STACK
 01DC E1            436         POP     H                   ;RESET STACK
 01DD D1            437         POP     D                   ;RESET STACK
 01DE C1            438         POP     B                   ;RESET STACK
 01DF FE00          439         CPI     00H                 ;  ON <NUL CHARACTER>
 01E1 C8            440         RZ                          ;  THEN RETURN
 01E2 03            441         INX     B                   ;NEXT LETTER
 01E3 C3D001    D   442         JMP     MSG                 ;CONTINUE
                    443 ;
                    444 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    445 ;
                    446 ;      ROUTINE NAME: BAUD
                    447 ;      INPUTS: C REG
                    448 ;      OUTPUTS: A REG
                    449 ;      CALLS: MSG
                    450 ;      DESCRIPTION: THIS ROUTINE TAKES THE ASCII VALUE FROM THE KEYBOARD
                    451 ;                   AND RETURNS THE CORRESPONDING BYTE TO MATCH THE REQUIRED
                    452 ;                   BAUD RATE.  THE ROUTINE ALSO PRINTS ON THE CONSOLE THE
                    453 ;                   REQUESTED BAUD RATE.
                    454 ;
                    455 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    456 ;
 01E6 79            457 BAUD:   MOV     A,C                 ;SET DATA IN ACCUMULATER
 01E7 FE31          458         CPI     31H                 ;INPUT 110 BAUD?
 01E9 CAFD01    D   459         JZ      BAUD1               ;IF YES THEN JUMP
 01EC FE32          460         CPI     32H                 ;INPUT 300 BAUD?
 01EE CA0602    D   461         JZ      BAUD3               ;IF YES THEN JUMP
 01F1 FE33          462         CPI     33H                 ;INPUT 1200 BAUD?
 01F3 CA0F02    D   463         JZ      BAUD12              ;IF YES THEN JUMP
 01F6 E1            464         POP     H                   ;RESET ADDRESS
 01F7 7D            465         MOV     A,L                 ;MOVE LOW H INTO A
 01F8 D60C          466         SUI     0CH                 ;SUBTRACT 9 STEPS
 01FA 6F            467         MOV     L,A                 ;MOVE LOW H BACK
 01FB E5            468         PUSH    H                   ;SET PROGRAM COUNTER
 01FC C9            469         RET                         ;RETURN TO PROMPT
 01FD 014C05    D   470 BAUD1:  LXI     B,MSG110            ;  PRINT BAUD SET
```

BEGIN LAST LINE OF TEXT
I OF AREA
E DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"

EPA-287 (Cin.)
(4-76)

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD; BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE    9

  LOC  OBJ          LINE        SOURCE STATEMENT

  0200 CDD001    D   471            CALL    MSG             ;  MESSAGE TO CONSOLE
  0203 3EAF          472            MVI     A,0AFH          ;MOVE DATA FOR 110 BAUD
  0205 C9            473            RET                     ;RETURN
  0206 017005    D   474 BAUD3:     LXI     B,MSG300        ;  PRINT BAUD SET
  0209 CDD001    D   475            CALL    MSG             ;  MESSAGE TO CONSOLE
  020C 3E40          476            MVI     A,40H           ;MOVE DATA FOR 300 BAUD TO A
  020E C9            477            RET                     ;RETURN
  020F 019405    D   478 BAUD12:    LXI     B,MSG120        ;  PRINT BAUD SET
  0212 CDD001    D   479            CALL    MSG             ;  MESSAGE TO CONSOLE
  0215 3E10          480            MVI     A,10H           ;MOVE DATA FOR 1200 BAUD TO A
  0217 C9            481            RET                     ;RETURN
                     482 ;
                     483 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     484 ;
                     485 ;       ROUTINE NAME: STCH
                     486 ;       INPUTS: E REG AND HL REG PAIR
                     487 ;       OUTPUTS: NONE
                     488 ;       CALLS: INCO
                     489 ;       DESCRIPTION: THIS ROUTINE STORES A PRESPECIFIED NUMBER OF CHARACTERS
                     490 ;                    STARTING AT A PRESPECIFIED MEMORY LOCATION.
                     491 ;
                     492 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     493 ;
  0218 1601          494 STCH:      MVI     D,01H           ;SET COUNT=1
  021A CDC001    D   495 STCH1:     CALL    INCO            ;GET CONSOLE INPUT
  021D 71            496            MOV     M,C             ;STORE IN MEMORY
  021E 7A            497            MOV     A,D             ;  CHECK NUMBER OF
  021F BB            498            CMP     E               ;  CHARACTERS ENTERED
  0220 C8            499            RZ                      ;RETURN IF FINISHED
  0221 23            500            INX     H               ;INCREMENT MEMORY
  0222 14            501            INR     D               ;INCREMENT COUNTER
  0223 C31A02    D   502            JMP     STCH1           ;CONTINUE
                     503 ;
                     504 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     505 ;
                     506 ;       ROUTINE NAME: MEMCHK
                     507 ;       INPUTS: NONE
                     508 ;       OUTPUTS: NONE
                     509 ;       CALLS: CO,MSG
                     510 ;       DESCRIPTION: THIS ROUTINE ZEROS OUT ALL MEMORY LOCATIONS AND BUFFER
                     511 ;                    AREA AND CHECKS FOR THE TOP OF MEMORY.
                     512 ;
                     513 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     514 ;
  0226 010000    E   515 MEMCHK:    LXI     B,HOME          ;SET HOME
  0229 CDD001    D   516            CALL    MSG             ;SEND TO CONSOLE
  022C 010000    E   517            LXI     B,TAB3          ;SET TAB
  022F CDD001    D   518            CALL    MSG             ;SEND TO CONSOLE
  0232 010000    E   519            LXI     B,UPRDY         ;SET LINE
  0235 CDD001    D   520            CALL    MSG             ;SEND TO CONSOLE
  0238 010000    E   521            LXI     B,UPRX          ;SET LINE
  023B CDD001    D   522            CALL    MSG             ;SEND TO CONSOLE
  023E 010000    E   523            LXI     B,UPSTBY        ;SET LINE
  0241 CDD001    D   524            CALL    MSG             ;SEND TO CONSOLE
  0244 014406    D   525            LXI     B,BUFFER        ;SET MESSAGE
  0247 CDD001    D   526            CALL    MSG             ;SEND TO CONSOLE
  024A 11203F        527            LXI     D,TOP           ;GET TOP ADDRESS REG
                     528            IF SBC EQ 00
                     529 ;
```

BEGIN LAST LINE OF TEXT
3/8"
PAGE NUMBER
DIMENSION FOR TABLES AND ILLUSTRATIONS
EPA-287 (Cin.)
(4-76)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   10

  LOC  OBJ         LINE        SOURCE STATEMENT

                    530 ;
                    531 ;          *****CHECK FOR 48K MEMORY (SKIP THIS FOR MDS-PURPOSES)*****
                    532 ;
                    533 ;
  024D 21FFFF       534 LONG:    LXI     H,BOTTOM+0BFFFH ;GET TOP OF 48K
  0250 7E           535          MOV     A,M             ;FETCH CONTENTS OF MEM
  0251 2F           536          CMA                     ;INVERT IT
  0252 77           537          MOV     M,A             ;ATTEMPT TO WRITE INTO MEM
  0253 BE           538          CMP     M               ;LOCATION READ/WRITE?
  0254 2F           539          CMA                     ;INVERT AGAIN
  0255 77           540          MOV     M,A             ;WRITE DATA BACK
  0256 C26702   D   541          JNZ     MID             ;IF NOT THEN JUMP
  0259 23           542          INX     H               ;INCREMENT MEMORY
  025A 0EB4         543          MVI     C,0B4H          ;SET DIGIT
  025C CD0703       544          CALL    CO              ;SEND TO CONSOLE
  025F 0EB8         545          MVI     C,0B8H          ;SET DIGIT
  0261 CD0703       546          CALL    CO              ;SEND TO CONSOLE
  0264 C37E02   D   547          JMP     SSET            ;JUMP
                    548          ENDIF
                    549 ;
                    550 ;
                    551 ;          *****CHECK FOR 32K MEMORY*****
                    552 ;
                    553 ;
  0267 21FFBF       554 MID:     LXI     H,BOTTOM+7FFFH  ;GET TOP OF 32 K
  026A 7E           555          MOV     A,M             ;FETCH CONTENTS OF MEM
  026B 2F           556          CMA                     ;INVERT IT
  026C 77           557          MOV     M,A             ;ATTEMPT TO WRITE INTO MEM
  026D BE           558          CMP     M               ;LOCATION READ/WRITE?
  026E 2F           559          CMA                     ;INVERT AGAIN
  026F 77           560          MOV     M,A             ;WRITE DATA BACK
  0270 C28C02   D   561          JNZ     SHORT           ;IF NOT THEN JUMP
  0273 23           562          INX     H               ;INCREMENT MEMORY
  0274 0EB3         563          MVI     C,0B3H          ;SET DIGIT
  0276 CD0703       564          CALL    CO              ;SEND TO CONSOLE
  0279 0EB2         565          MVI     C,0B2H          ;SET DIGIT
  027B CD0703       566          CALL    CO              ;SEND TO CONSOLE
                    567 ;
                    568 ;
                    569 ;          *****STORE SIZE OF MEMORY IN 'TOP'*****
                    570 ;
                    571 ;
  027E 7C           572 SSET:    MOV     A,H             ;SET HIGH BYTE IN ACCUM
  027F 12           573          STAX    D               ;SET IN REGISTER
  0280 13           574          INX     D               ;INCREMENT REGISTER
  0281 7D           575          MOV     A,L             ;SET LOW BYTE IN ACCUM
  0282 12           576          STAX    D               ;SET IN REGISTER
  0283 015606   D   577          LXI     B,BLKS          ;SET MESSAGE
  0286 CDD001   D   578          CALL    MSG             ;SEND TO CONSOLE
  0289 C39C02   D   579          JMP     ZERO            ;JUMP
                    580 ;
                    581 ;
                    582 ;          *****ASSSUME 16K MEMORY*****
                    583 ;
                    584 ;
  028C 21007C       585 SHORT:   LXI     H,BOTTOM+3C00H  ;GET TOP OF 16 K
  028F 0EB1         586          MVI     C,0B1H          ;SET DIGIT
  0291 CD0703       587          CALL    CO              ;SEND TO CONSOLE
  0294 0EB6         588          MVI     C,0B6H          ;SET DIGIT
```

BEGIN FIRST LINE OF TEXT HERE
TYPING GUIDE SHEET
CENTER OF PAGE
TOP OF IMAGE
DROPPED HEAD BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   11

 LOC  OBJ          LINE        SOURCE STATEMENT

 0296 CD0703        589          CALL    CO             ;SEND TO CONSOLE
 0299 C37E02   D    590          JMP     SSET           ;JUMP
                    591 ;
                    592 ;
                    593 ;        *****ZERO THE ENTIRE MEMORY*****
                    594 ;
                    595 ;
 029C 21103F        596 ZERO:    LXI     H,SYSTAT       ;SET SYSTEM STATUS IN HL
 029F 3600          597          MVI     M,00H          ;SET RUN CODE IN MEM
 02A1 23            598          INX     H              ;GET OVRFLW
 02A2 3600          599          MVI     M,00H          ;RESET OVRFLW
 02A4 21103C        600          LXI     H,STAT         ;LOAD USER STATUS
 02A7 3E01          601          MVI     A,01H          ;  SET USER 0
 02A9 77            602          MOV     M,A            ;  TO READY
 02AA 21103D        603          LXI     H,STAT+100H    ;    SET USER 1
 02AD 77            604          MOV     M,A            ;    TO READY
 02AE 21103E        605          LXI     H,STAT+200H    ;  SET USER 2
 02B1 77            606          MOV     M,A            ;  TO READY
 02B2 21113C        607          LXI     H,SEQ0         ;LOAD SEQUENCE COUNTER
 02B5 CDDB02   D    608          CALL    ZERO1          ;CALL ZERO ROUTINE
 02B8 32153C        609          STA     STBY0          ;ZERO STBY COUNTER
 02BB 21113D        610          LXI     H,SEQ0+100H    ;LOAD SEQUENCE COUNTER 1
 02BE CDDB02   D    611          CALL    ZERO1          ;CALL ZERO ROUTINE
 02C1 32153D        612          STA     STBY1          ;ZERO STBY COUNTER
 02C4 21113E        613          LXI     H,SEQ0+200H    ;LOAD SEQUENCE COUNTER 2
 02C7 CDDB02   D    614          CALL    ZERO1          ;CALL ZERO ROUTINE
 02CA 32153E        615          STA     STBY2          ;ZERO STBY COUNTER
 02CD 210040        616          LXI     H,BOTTOM       ;LOAD BOTTOM OF MEMORY
 02D0 3A203F        617 CONT:    LDA     TOP            ;GET TOP OF MEMORY
 02D3 BC            618          CMP     H              ;AT TOP
 02D4 C8            619          RZ                     ;IF YES THEN RETURN
 02D5 AF            620          XRA     A              ;ZERO ACCUM
 02D6 77            621          MOV     M,A            ;SET LOCATION TO ZERO
 02D7 23            622          INX     H              ;INCREMENT MEMORY
 02D8 C3D002   D    623          JMP     CONT           ;JUMP
 02DB AF            624 ZERO1:   XRA     A              ;ZERO ACCUMULATER
 02DC 77            625          MOV     M,A            ;SET COUNTER TO ZERO
 02DD 23            626          INX     H              ;LOAD BLOCK COUNTER 0
 02DE 77            627          MOV     M,A            ;SET COUNTER TO ZERO
 02DF 23            628          INX     H              ;LOAD BLOCK POINTER 0
 02E0 77            629          MOV     M,A            ;SET COUNTER TO ZERO
 02E1 23            630          INX     H              ;LOAD BLOCK POINTER 0
 02E2 77            631          MOV     M,A            ;SET COUNTER TO ZERO
 02E3 23            632          INX     H              ;GET END COUNTER
 02E4 23            633          INX     H              ;LOAD END COUNTER 0 LSB
 02E5 77            634          MOV     M,A            ;SET COUNTER TO ZERO
 02E6 23            635          INX     H              ;LOAD END COUNTER 0 MSB
 02E7 77            636          MOV     M,A            ;SET COUNTER TO ZERO
 02E8 C9            637          RET                    ;RETURN
                    638 ;
                    639 ;
                    640 ;        *****CONSTANTS*****
                    641 ;
                    642 ;
 02E9 1B            643 TITLE:   DB      1BH,48H,1BH,4AH,0AH
 02EA 48
 02EB 1B
 02EC 4A
 02ED 0A
```

BEGIN LAST LINE OF TEXT
3/8"
M OF AREA; DE DIMENSION FOR TABLES AND ILLUS-TRATIONS

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   12

  LOC  OBJ         LINE        SOURCE STATEMENT

  02EE 20202020     644 TITLE1: DB    '           ENVIRONMENTAL MONITORING AND SUPPORT LABORATORY - CINCIN
                        NATI',0AH,0DH
  02F2 20202020
  02F6 2020454E
  02FA 5649524F
  02FE 4E4D454E
  0302 54414C20
  0306 4D4F4E49
  030A 544F5249
  030E 4E472041
  0312 4E442053
  0316 5550504F
  031A 5254204C
  031E 41424F52
  0322 41544F52
  0326 59202020
  032A 43494E43
  032E 494E4E41
  0332 5449
  0334 0A
  0335 0D
  0336 20202020     645 TITLE2: DB    '                          ADVANCED INSTRUMENTATION SECTION',0AH,0AH,
                        0DH
  033A 20202020
  033E 20202020
  0342 20202020
  0346 20202020
  034A 20202020
  034E 41445641
  0352 4E434544
  0356 20494E53
  035A 5452554D
  035E 454E5441
  0362 54494F4E
  0366 20534543
  036A 54494F4E
  036E 0A
  036F 0A
  0370 0D
  0371 20202020     646 TITLE3: DB    '                           MICROPROCESSOR BUFFER SYSTEM',0AH,0DH
  0375 20202020
  0379 20202020
  037D 20202020
  0381 20202020
  0385 20202020
  0389 20204D49
  038D 43524F50
  0391 524F4345
  0395 53534F52
  0399 20425546
  039D 46455220
  03A1 53595354
  03A5 454D
  03A7 0A
  03A8 0D
  03A9 20202020     647 TITLE4: DB    '                                   VERSION  2.1',0AH,0AH,0DH,00H
  03AD 20202020
  03B1 20202020
  03B5 20202020
```

EPA-287 (Cin.)
(4-76)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   13

  LOC  OBJ          LINE        SOURCE STATEMENT

  03B9 20202020
  03BD 20202020
  03C1 20202020
  03C5 20202020
  03C9 20205645
  03CD 5253494F
  03D1 4E202032
  03D5 2E31
  03D7 0A
  03D8 0A
  03D9 0D
  03DA 00
  03DB 1B            648 MEMLK:  DB     1BH,6CH,00H
  03DC 6C
  03DD 00
  03DE 1B            649 MEMULK: DB     1BH,6DH,00H
  03DF 6D
  03E0 00
  03E1 504F5254      650 PRMT0:  DB     'PORT 0.....',0AH,0DH,0AH
  03E5 20302E2E
  03E9 2E2E2E
  03EC 0A
  03ED 0D
  03EE 0A
  03EF 20202020      651 PRMT01: DB     '          SPEEDS:   1 = 110 BAUD',0AH,0DH
  03F3 20202020
  03F7 20205350
  03FB 45454453
  03FF 3A202020
  0403 31203D20
  0407 31313020
  040B 42415544
  040F 0A
  0410 0D
  0411 20202020      652 PRMT02: DB     '                    2 = 300 BAUD',0AH,0DH
  0415 20202020
  0419 20202020
  041D 20202020
  0421 20202020
  0425 32203D20
  0429 33303020
  042D 42415544
  0431 0A
  0432 0D
  0433 20202020      653 PRMT03: DB     '                    3 =1200 BAUD',0AH,0DH,0AH
  0437 20202020
  043B 20202020
  043F 20202020
  0443 20202020
  0447 33203D31
  044B 32303020
  044F 42415544
  0453 0A
  0454 0D
  0455 0A
  0456 20202020      654 ENTSP:  DB     '     ENTER SPEED (1, 2, OR 3): ',00H
  045A 20454E54
  045E 45522053
  0462 50454544
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

0466 2028312C
046A 20322C20
046E 4F522033
0472 293A20
0475 00
0476 0A            655 STRTCH: DB      0AH,0AH,0DH,'     ENTER FIVE STARTING CHARACTERS: ',00H
0477 0A
0478 0D
0479 20202020
047D 20454E54
0481 45522046
0485 49564520
0489 53544152
048D 54494E47
0491 20434841
0495 52414354
0499 4552533A
049D 20
049E 00
049F 0A            656 STOPCH: DB      0AH,0AH,0DH,'     ENTER FIVE STOPPING CHARACTERS: ',00H
04A0 0A
04A1 0D
04A2 20202020
04A6 20454E54
04AA 45522046
04AE 49564520
04B2 53544F50
04B6 50494E47
04BA 20434841
04BE 52414354
04C2 4552533A
04C6 20
04C7 00
04C8 0A            657 ACCTID: DB      0AH,0AH,0DH,'     ENTER ACCOUNT ID: ',00H
04C9 0A
04CA 0D
04CB 20202020
04CF 20454E54
04D3 45522041
04D7 43434F55
04DB 4E542049
04DF 443A20
04E2 00
04E3 0A            658 SETMSG: DB      0AH,0AH,0AH,0AH,0AH,0AH,0AH,0AH
04E4 0A
04E5 0A
04E6 0A
04E7 0A
04E8 0A
04E9 0A
04EA 0A
04EB 0A            659         DB      0AH,0AH,0AH,0AH,0AH,0AH,0AH,0AH
04EC 0A
04ED 0A
04EE 0A
04EF 0A
04F0 0A
04F1 0A
04F2 0A
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   15

  LOC  OBJ           LINE        SOURCE STATEMENT

  04F3 0A             660          DB      0AH,1BH,41H,1BH,41H,1BH,41H,1BH
  04F4 1B
  04F5 41
  04F6 1B
  04F7 41
  04F8 1B
  04F9 41
  04FA 1B
  04FB 41             661          DB      41H,1BH,41H,1BH,41H,1BH,41H,1BH
  04FC 1B
  04FD 41
  04FE 1B
  04FF 41
  0500 1B
  0501 41
  0502 1B
  0503 41             662          DB      41H,1BH,41H,1BH,41H,1BH,41H,1BH
  0504 1B
  0505 41
  0506 1B
  0507 41
  0508 1B
  0509 41
  050A 1B
  050B 41             663          DB      41H,1BH,41H,1BH,41H,1BH,41H,1BH
  050C 1B
  050D 41
  050E 1B
  050F 41
  0510 1B
  0511 41
  0512 1B
  0513 41             664          DB      41H,00H
  0514 00
  0515 0D             665 PRMT1:   DB      0DH,'PORT 1.....',0AH,0DH,0AH,00H
  0516 504F5254
  051A 20312E2E
  051E 2E2E2E
  0521 0A
  0522 0D
  0523 0A
  0524 00
  0525 0D             666 PRMT2:   DB      0DH,'PORT 2.....',0AH,0DH,0AH,00H
  0526 504F5254
  052A 20322E2E
  052E 2E2E2E
  0531 0A
  0532 0D
  0533 0A
  0534 00
  0535 0D             667 PRMT3:   DB      0DH,'PORT 3 (NOVA).....',0AH,0DH,0AH,00H
  0536 504F5254
  053A 20332028
  053E 4E4F5641
  0542 292E2E2E
  0546 2E2E
  0548 0A
  0549 0D
  054A 0A
```

BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA

```
 LOC  OBJ         LINE       SOURCE STATEMENT

 054B 00
 054C 0A           668 MSG110: DB      0AH,0AH,0DH,'          PORT SET TO 110 BAUD',0AH,0DH,00H
 054D 0A
 054E 0D
 054F 20202020
 0553 20202020
 0557 2020504F
 055B 52542053
 055F 45542054
 0563 4F203131
 0567 30204241
 056B 5544
 056D 0A
 056E 0D
 056F 00
 0570 0A           669 MSG300: DB      0AH,0AH,0DH,'          PORT SET TO 300 BAUD',0AH,0DH,00H
 0571 0A
 0572 0D
 0573 20202020
 0577 20202020
 057B 2020504F
 057F 52542053
 0583 45542054
 0587 4F203330
 058B 30204241
 058F 5544
 0591 0A
 0592 0D
 0593 00
 0594 0A           670 MSG120: DB      0AH,0AH,0DH,'          PORT SET TO 1200 BAUD',0AH,0DH,00H
 0595 0A
 0596 0D
 0597 20202020
 059B 20202020
 059F 2020504F
 05A3 52542053
 05A7 45542054
 05AB 4F203132
 05AF 30302042
 05B3 415544
 05B6 0A
 05B7 0D
 05B8 00
 05B9 20202020     671 HEADER: DB      '          USER 0    USER 1    USER 2',0AH,0AH,0DH
 05BD 20202020
 05C1 20205553
 05C5 45522030
 05C9 20202020
 05CD 55534552
 05D1 20312020
 05D5 20205553
 05D9 45522032
 05DD 0A
 05DE 0A
 05DF 0D
 05E0 20755020     672         DB      ' uP READY',0AH,0AH,0DH
 05E4 52454144
 05E8 59
 05E9 0A
```

BEGIN LAST LINE OF TEXT

PAGE NUMBER

TYPING GUIDE SHEET

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   17

 LOC  OBJ          LINE        SOURCE STATEMENT

 05EA 0A
 05EB 0D
 05EC 20755020      673         DB      ' uP RCVG',0AH,0AH,0DH
 05F0 52435647
 05F4 0A
 05F5 0A
 05F6 0D
 05F7 20755020      674         DB      ' uP STBY',0AH,0AH,0DH
 05FB 53544259
 05FF 0A
 0600 0A
 0601 0D
 0602 20755020      675         DB      ' uP TRANS',00H
 0606 5452414E
 060A 53
 060B 00
 060C 1B            676 TABDAT: DB      1BH,48H,'          '
 060D 48
 060E 20202020
 0612 20202020
 0616 20202020
 061A 2020
 061C 1B            677         DB      1BH,31H,'          ',1BH,31H
 061D 31
 061E 20202020
 0622 20202020
 0626 2020
 0628 1B
 0629 31
 062A 20202020      678         DB      '          ',1BH,31H
 062E 20202020
 0632 2020
 0634 1B
 0635 31
 0636 20202020      679         DB      '           ',1BH,31H,00H
 063A 20202020
 063E 202020
 0641 1B
 0642 31
 0643 00
 0644 0A            680 BUFFER: DB      0AH,'  BUFFER SIZE:  ',00H
 0645 20204255
 0649 46464552
 064D 2053495A
 0651 453A2020
 0655 00
 0656 20424C4F      681 BLKS:   DB      ' BLOCKS',00H
 065A 434B53
 065D 00
 065E 0A            682 TIMER:  DB      0AH,0DH,'TIMER INTERRUPT LEVEL 5',00H
 065F 0D
 0660 54494D45
 0664 5220494E
 0668 54455252
 066C 55505420
 0670 4C455645
 0674 4C2035
 0677 00
 0000          D    683         END     SETUP
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        SETUP    PAGE   18

  LOC  OBJ         LINE        SOURCE STATEMENT


PUBLIC SYMBOLS
BASE   A 3C00    BLKO   A 3C12    BLK1   A 3D12    BLK2   A 3E12    BOTTOM A 4000    CHAAMT A 3F31
CHAPTR A 3F13    CI     A 02F4    CO     A 0307    CURBLK A 3F1F    FSTBLK A 3F30    LSTUSR A 3F15
MSG    D 01D0    NOINST A 3F16    NOVSTS A 3F17    OVRFLW A 3F11    PIT5   A 00B0    ROUPTR A 3F18
RUNFLG A 3F33    SENT0  A 3C18    SEQ0   A 3C11    SEQ1   A 3D11    SEQ2   A 3E11    SNTP1  A 3F1A
STAT   A 3C10    STBY0  A 3C15    STBY1  A 3D15    STBY2  A 3E15    SYSTAT A 3F10    TIMCT1 A 3F1C
TIMCTR A 3F1B    TOP    A 3F20    TTYBUF A 3CB0    UARTLI A 0027    UARTLO A 0025    USRFLG A 3F1E
VECFLG A 3F34    ZERO1  D 02DB

EXTERNAL SYMBOLS
EOI    E 0000    HOME   E 0000    TAB3   E 0000    TB534  E 0000    UPRDY  E 0000    UPRX   E 0000
UPSTBY E 0000    WAIT   E 0000

USER SYMBOLS
ACCTID D 04C8    BASE   A 3C00    BAUD   D 01E6    BAUD1  D 01FD    BAUD12 D 020F    BAUD3  D 0206
BLKO   A 3C12    BLK1   A 3D12    BLK2   A 3E12    BLKS   D 0656    BOTTOM A 4000    BUFFER D 0644
CHAAMT A 3F31    CHAPTR A 3F13    CI     A 02F4    CO     A 0307    CONT   D 02D0    CURBLK A 3F1F
ENTSP  D 0456    EOI    E 0000    FSTBLK A 3F30    HEADER D 05B9    HOME   E 0000    INCO   D 01C0
INT4   A 3FF0    JPTB   D 01AB    LOAD   D 0194    LONG   D 024D    LSTUSR A 3F15    MEMCHK D 0226
MEMLK  D 03DB    MEMULK D 03DE    MID    D 0267    MSG    D 01D0    MSG110 D 054C    MSG120 D 0594
MSG300 D 0570    NOINST A 3F16    NOVSTS A 3F17    NXTBYT D 0019    OVRFLW A 3F11    PIT03  A 0016
PIT14  A 0056    PIT25  A 0096    PIT4   A 0076    PIT5   A 00B0    PRMT0  D 03E1    PRMT01 D 03EF
PRMT02 D 0411    PRMT03 D 0433    PRMT1  D 0515    PRMT2  D 0525    PRMT3  D 0535    ROUPTR A 3F18
RUNFLG A 3F33    SBC    A 0000    SENT0  A 3C18    SEQ0   A 3C11    SEQ1   A 3D11    SEQ2   A 3E11
SETMSG D 04E3    SETUP  D 0000    SHORT  D 028C    SNTP1  A 3F1A    SSET   D 027E    STAT   A 3C10
STBY0  A 3C15    STBY1  A 3D15    STBY2  A 3E15    STCH   D 0218    STCH1  D 021A    STOPCH D 049F
STRTCH D 0476    SYSTAT A 3F10    TAB3   E 0000    TABDAT D 060C    TB534  E 0000    TIMCT1 A 3F1C
TIMCTR A 3F1B    TIMER  D 065E    TIMOUT D 01B3    TITLE  D 02E9    TITLE1 D 02EE    TITLE2 D 0336
TITLE3 D 0371    TITLE4 D 03A9    TOP    A 3F20    TTYBUF A 3CB0    UARTLI A 0027    UARTLO A 0025
UARTMO A 004F    UPRDY  E 0000    UPRX   E 0000    UPSTBY E 0000    USRFLG A 3F1E    VECFLG A 3F34
WAIT   E 0000    ZERO   D 029C    ZERO1  D 02DB

ASSEMBLY COMPLETE,   NO ERRORS
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                    PAGE    1

ACCTID   168     201     234     657#
BASE      39      48#     60      61      64      67      73      79
BAUD     156     189     222     253     457#
BAUD1    459     470#
BAUD12   463     478#
BAUD3    461     474#
BLK0      40      61#     62      63
BLK1      40      62#
BLK2      40      63#
BLKS     577     681#
BOTTOM    39      59#    534     554     585     616
BUFFER   525     680#
CHAAMT    41      91#     92
CHAPTR    41      80#     81     120
CI        39      47#    405
CO        39      46#    408     434     544     546     564     566     587     589
CONT     617#    623
CURBLK    42      89#
ENTSP    186     219     250     654#
EOI       43     389
FSTBLK    42      90#     91     127
HEADER   319     671#
HOME      43     515
INCO     155     188     221     252     402#    495
INT4      49#    324
JPTB     323     357#
LOAD     339#    344
LONG     534#
LSTUSR    41      81#     82
MEMCHK   349     515#
MEMLK    111     648#
MEMULK   315     649#
MID      541     554#
MSG       39     110     112     154     159     164     169     176     185     187     192     197     202     209
         218     220     225     230     235     242     249     251     256     316     318     320     322     388
         427#    442     471     475     479     516     518     520     522     524     526     578
MSG110   470     668#
MSG120   478     670#
MSG300   474     669#
NOINST    41      82#     83
NOVSTS    41      83#     84
NXTBYT   122#    126
OVRFLW    40      72#
PIT03     74#    150
PIT14     75#    182
PIT25     76#    215
PIT4      77#    262
PIT5      39      78#
PRMT0    153     650#
PRMT01   651#
PRMT02   652#
PRMT03   653#
PRMT1    184     665#
PRMT2    217     666#
PRMT3    248     667#
ROUPTR    42      84#     85
RUNFLG    42      92#     93
SBC       44#     45     103     302     325     359     367     528
SENT0     41      79#    137
SEQ0      40      64#     65      66     607     610     613
SEQ1      40      65#
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                PAGE   2

SEQ2     40      66#
SETMSG  175     208     241     255     658#
SETUP    38     103#    683
SHORT   561     585#
SNTP1    42      85#     86
SSET    547     572#    590
STAT     39      60#     70     600     603     605
STBY0    40      67#     68      69     609
STBY1    40      68#    612
STBY2    40      69#    615
STCH    162     167     172     195     200     205     228     233     238     494#
STCH1   495#    502
STOPCH  163     196     229     656#
STRTCH  158     191     224     655#
SYSTAT   39      70#     71      72      80      90     596
TAB3     43     517
TABDAT  321     676#
TB534    43     358
TIMCT1   42      87#     88
TIMCTR   42      86#     87
TIMER   387     682#
TIMOUT  366     383#
TITLE   109     317     643#
TITLE1  644#
TITLE2  645#
TITLE3  646#
TITLE4  647#
TOP      39      71#    527     617
TTYBUF   39      73#    160     165     170     193     198     203     226     231     236
UARTLI   39      57#    279
UARTLO   39      58#    283
UARTMO   56#    274
UPRDY    43     519
UPRX     43     521
UPSTBY   43     523
USRFLG   42      88#     89     143
VECFLG   41      93#
WAIT     43     351
ZERO    579     596#
ZERO1    40     608     611     614     624#

CROSS REFERENCE COMPLETE
```

7.2 TB534

TB534 is the module which gets control of the microcomputer immediately whenever any interrupt occurs. When one of the UARTs on the iSBC 534 Board receives a character, it sets an interrupt on PICO of that board. A character arriving on UART0 sets interrupt 0 on PICO. UART1 sets interrupt 2, UART2 sets interrupt 4, and UART3 sets interrupt 6. When any interrupt is set on PICO, PICO sets interrupt 4 of the programmable interrupt controller on the iSBC 80/20-4 Single Board Computer. That interrupt causes control to jump to TB534, as specified by the interrupt jump table put in place by the SETUP Module. Then, it is the responsibility of TB534 to determine which UART caused the interrupt and to branch to the appropriate routine to service that interrupt.

TB534 first saves the current values of all the registers in the stack. It then writes the Operation Command Word OC_H to PICO of the iSBC 534 Board at I/O address 18_H. Reading from the same address then causes PICO to provide the binary coded decimal expression for the number of the highest level requesting interrupt. TB534 then calls one of four subroutines. USER0, USER1, and USER2 are subroutines which intercept characters arriving from the intelligent instruments. NOVA is the name of the subroutine which handles characters arriving from the NOVA computer.

When the subroutine has processed the new character and returned control to TB534, this module resets the programmable interrupt controllers. Writing a 20_H to I/O address DA_H resets the PIC on the iSBC 80/20-4 board. Writing an $A0_H$ to I/O address 18_H resets PICO on the iSBC 534 Board.

Finally, TB534 restores the values of the registers from the stack, enables the interrupts, and returns to the procedure which was executing at the time of the interrupt.

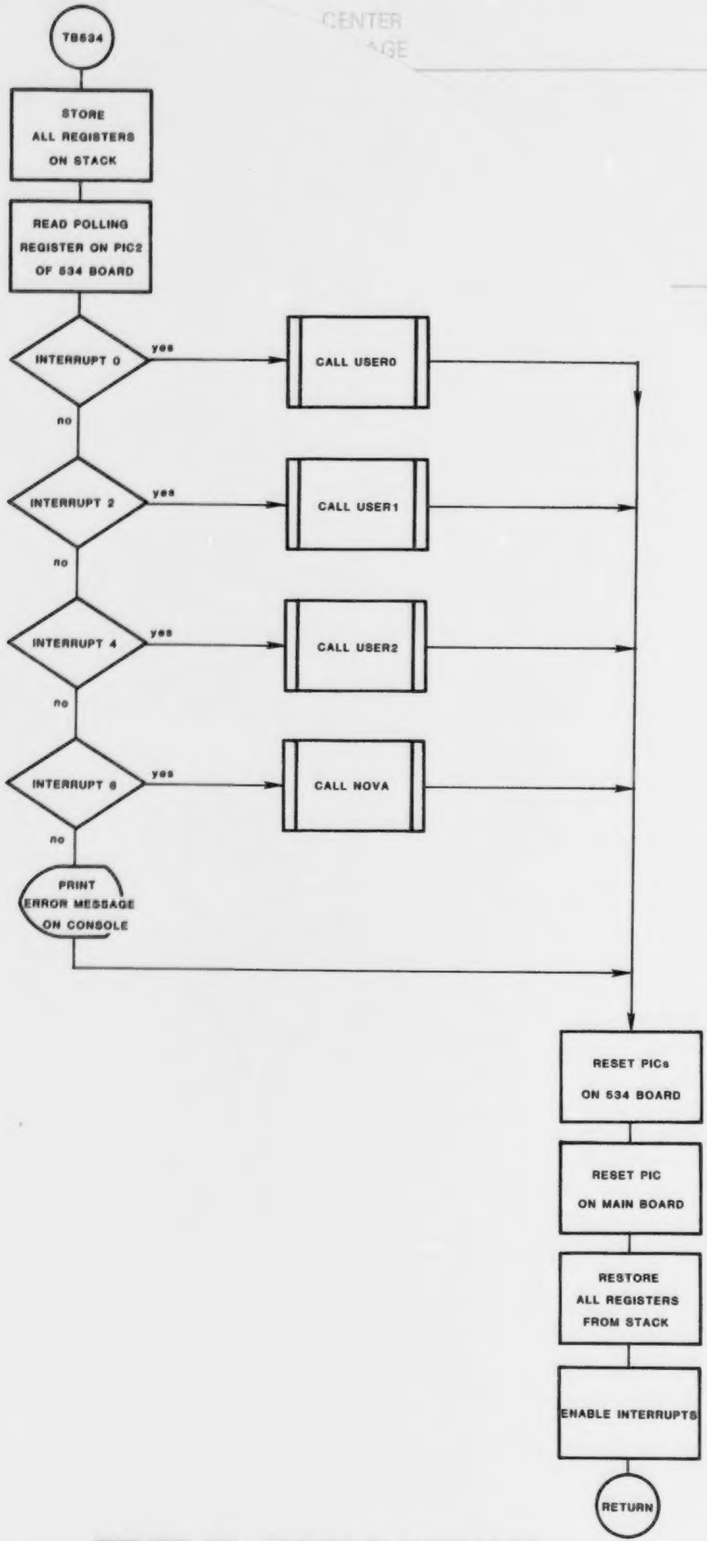

TB534
STORE ALL REGISTERS ON STACK
READ POLLING REGISTER ON PIC2 OF 534 BOARD
INTERRUPT 0
yes
CALL USER0
no
INTERRUPT 2
yes
CALL USER1
no
INTERRUPT 4
yes
CALL USER2
no
INTERRUPT 6
yes
CALL NOVA
no
PRINT ERROR MESSAGE ON CONSOLE
RESET PICs ON 534 BOARD
RESET PIC ON MAIN BOARD
RESTORE ALL REGISTERS FROM STACK
ENABLE INTERRUPTS
RETURN

FIGURE 10. TB534 FLOWCHART.

BEGIN FIRST LINE OF TEXT HERE
DROPPED HEAD; BEGIN SECTIONS HERE
BEGIN LAST LINE OF TEXT
TOP OF IMAGE AREA
BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

```
ASM80 :F1:TB534.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        TB534    PAGE    1


  LOC  OBJ         LINE       SOURCE STATEMENT

                      1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      2 ;
                      3 ;         MODULE NAME:  TB534
                      4 ;
                      5 ;         DESCRIPTION:  THIS MODULE DETERMINES THE INTERRUPT VECTORS FOR
                      6 ;                       THE EXTERNAL INTERRUPT CONTROLLERS ON THE SBC534
                      7 ;                       BOARD.  IT ALSO CONTAINS THE END OF INTERRUPT
                      8 ;                       SUBROUTINES.
                      9 ;
                     10 ;         MOST RECENT REVISION:  AUGUST 22, 1980
                     11 ;
                     12 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     13 ;
                     14         NAME    TB534
                     15         PUBLIC  EOI,EOIM,EOI534,TB534
                     16         EXTRN   NOVA,MSG,USER0,USER1,USER2
  0000               17 SBC     SET     00H                 ;CONDITIONAL ASSEMBLY FLAG (00-SBC FF-MDS)
                     18         DSEG                        ;CONTINUE IN DATA SEGMENT
                     19 ;
                     20 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     21 ;
                     22 ;      ROUTINE NAME: TB534
                     23 ;
                     24 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     25 ;
  0000 F5            26 TB534:  PUSH    PSW                 ;SAVE REGISTER
  0001 C5            27         PUSH    B                   ;SAVE REGISTER
  0002 D5            28         PUSH    D                   ;SAVE REGISTER
  0003 E5            29         PUSH    H                   ;SAVE REGISTER
  0004 3E0C          30         MVI     A,0CH               ;  READ POLLING
  0006 D318          31         OUT     18H                 ;  REGISTER ON
  0008 DB18          32         IN      18H                 ;  PIC0
  000A E607          33         ANI     07H                 ;ISOLATE BITS
  000C FE00          34         CPI     00H                 ;CHECK INTERRUPT 0
  000E CA2900   D    35         JZ      TABLE               ;IF SO JUMP
  0011 FE02          36         CPI     02H                 ;CHECK INTERRUPT 2
  0013 CA2F00   D    37         JZ      TABLE+06H           ;IF SO JUMP
  0016 FE04          38         CPI     04H                 ;CHECK INTERRUPT 4
  0018 CA3500   D    39         JZ      TABLE+0CH           ;IF SO JUMP
  001B FE06          40         CPI     06H                 ;CHECK INTERRUPT 6
  001D CA3B00   D    41         JZ      TABLE+12H           ;IF SO JUMP
  0020 015600   D    42         LXI     B,NOUSER            ;  SEND ERROR TO
  0023 CD0000   E    43         CALL    MSG                 ;  CONSOLE
  0026 C33E00   D    44         JMP     EOI                 ;END OF INTERRUPT
  0029 CD0000   E    45 TABLE:  CALL    USER0               ;SERVICE USER 0
  002C C33E00   D    46         JMP     EOI                 ;JUMP
  002F CD0000   E    47         CALL    USER1               ;SERVICE USER 1
  0032 C33E00   D    48         JMP     EOI                 ;JUMP
  0035 CD0000   E    49         CALL    USER2               ;SERVICE USER 2
  0038 C33E00   D    50         JMP     EOI                 ;JUMP
  003B CD0000   E    51         CALL    NOVA                ;SERVICE NOVA
                     52 ;
                     53 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     54 ;
                     55 ;      ROUTINE NAME: EOI
                     56 ;      INPUTS: NONE
                     57 ;      OUTPUTS: NONE
                     58 ;      CALLS: EOI534,EOIM
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        TB534     PAGE    2

  LOC  OBJ         LINE        SOURCE STATEMENT

                     59 ;      DESCRIPTION: THIS SUBROUTINE IS JUMPED TO RATHER THAN CALLED AT THE
                     60 ;                   END OF AN INTERRUPT.  IT CAUSES THE SYSTEM TO RESUME
                     61 ;                   FROM THE PLACE IT WAS INTERRUPTED.
                     62 ;
                     63 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     64 ;
  003E CD4F00   D    65 EOI:     CALL    EOI534           ;EOI TO 534 CARD
  0041 CD4A00   D    66          CALL    EOIM             ;EOI TO MAIN BOARD
  0044 E1            67          POP     H                ;RESTORE REGISTER
  0045 D1            68          POP     D                ;RESTORE REGISTER
  0046 C1            69          POP     B                ;RESTORE REGISTER
  0047 F1            70          POP     PSW              ;RESTORE REGISTER
  0048 FB            71          EI                       ;ENABLE INTERRUPTS
  0049 C9            72          RET                      ;RESUME TASK
  004A 3E20          73 EOIM:    MVI     A,20H            ;EOI
                     74          IF SBC EQ 0FFH
                     75          OUT     0FDH             ;MAIN BOARD (MDS-800)
                     76          ELSE
  004C D3DA          77          OUT     0DAH             ;MAIN BOARD (SBC-80/20-4)
                     78          ENDIF
  004E C9            79          RET                      ;RETURN
  004F 3EA0          80 EOI534:  MVI     A,0A0H           ;EOI
  0051 D318          81          OUT     18H              ;PIC 0
  0053 D31A          82          OUT     1AH              ;PIC 1
  0055 C9            83          RET                      ;RETURN
  0056 0A            84 NOUSER:  DB      0AH,0DH,'NO USER - ERROR',00H
  0057 0D
  0058 4E4F2055
  005C 53455220
  0060 2D204552
  0064 524F52
  0067 00
                     85          END                      ;END OF MODULE

PUBLIC SYMBOLS
EOI    D 003E    EOI534 D 004F    EOIM   D 004A    TB534  D 0000

EXTERNAL SYMBOLS
MSG    E 0000    NOVA   E 0000    USER0  E 0000    USER1  E 0000    USER2  E 0000

USER SYMBOLS
EOI    D 003E    EOI534 D 004F    EOIM   D 004A    MSG    E 0000    NOUSER D 0056    NOVA   E 0000
SBC    A 0000    TABLE  D 0029    TB534  D 0000    USER0  E 0000    USER1  E 0000    USER2  E 0000

ASSEMBLY COMPLETE,   NO ERRORS
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                  PAGE   1

EOI      15      44      46      48      50      65#
EOI534   15      65      80#
EOIM     15      66      73#
MSG      16      43
NOUSER   42      84#
NOVA     16      51
SBC      17#     74
TABLE    35      37      39      41      45#
TB534    14      15      26#
USER0    16      45
USER1    16      47
USER2    16      49

CROSS REFERENCE COMPLETE
```

7.3 WAIT

WAIT is actually the main module of the Microprocessor Buffer System, although all it does is update the interface box lights, call as subroutines other modules larger than itself, and allow itself to be interrupted by arriving characters. Thus, the only real task of WAIT is to get the User Status Words from memory, set their "error" and "stop data" bits if mandated to do so by the System Status Word, and then send the User Status Words to the interface boxes in the laboratories, where they are decoded to light the proper LED indicators. The configuration of the User Status Words is shown in Figure 11, and that of the System Status Word is shown in Figure 12.

As WAIT begins, it examines the System Status Word, SYSTAT, to determine if any error is currently effecting the system. As shown in Figure 12, a 1 in any of the low order bits of SYSTAT indicates an error. Only the memory overflow error effects the ability of the system to receive more characters from the intelligent instruments, however. Therefore, WAIT recognizes one of three possible situations: 1) no error is in effect, 2) a non-fatal error is in effect, or 3) a fatal error is in effect.

WAIT then gets STAT0, the User Status Word for User 0, from memory. The "ready," "receiving," "standby," and/or "transmitting" bits are already set properly by other parts of the system. WAIT sets the "error" bit if any system error is in effect and sets the "stop data" bit if the error is a memory overflow fatal error. It then writes STAT0 to UART0 at I/O address 10_H. This transmits the User Status Word to the interface box for user 0. Similarly, WAIT gets the User Status Words STAT1 and STAT2, sets their "error" and "stop data" bits (if necessary), and writes them to UART1 and UART2 at I/O addresses 12_H and 14_H.

Finally, WAIT calls the OPER module as a subroutine, and then starts over. WAIT is an endless loop which executes about every 1.6 seconds.

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD, BEGIN SECTIONS HERE

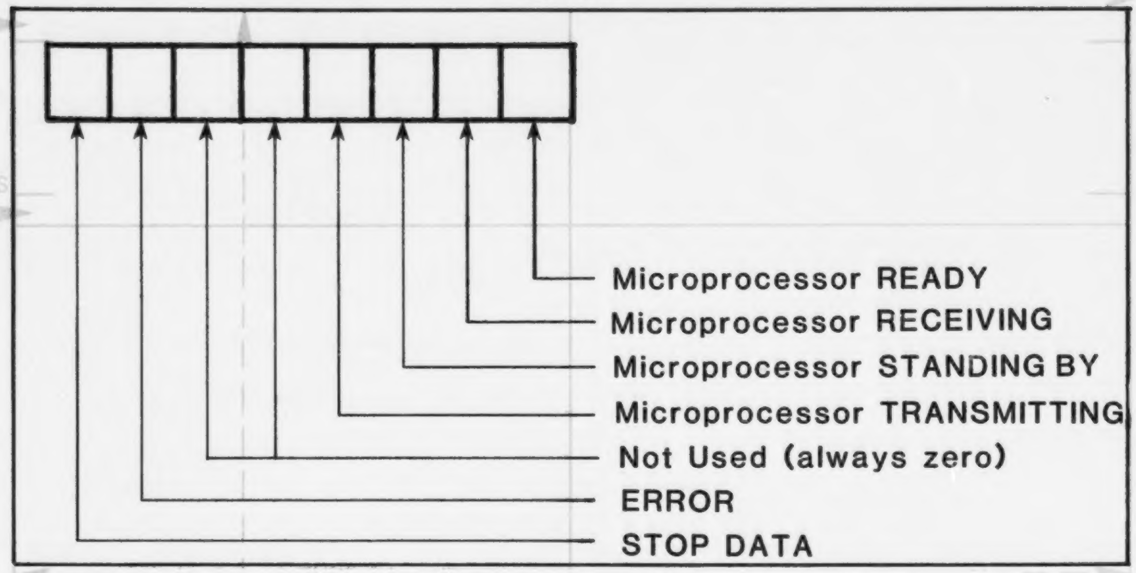

Microprocessor READY
Microprocessor RECEIVING
Microprocessor STANDING BY
Microprocessor TRANSMITTING
Not Used (always zero)
ERROR
STOP DATA

FIGURE 11. USER STATUS WORD.

9-1/8"

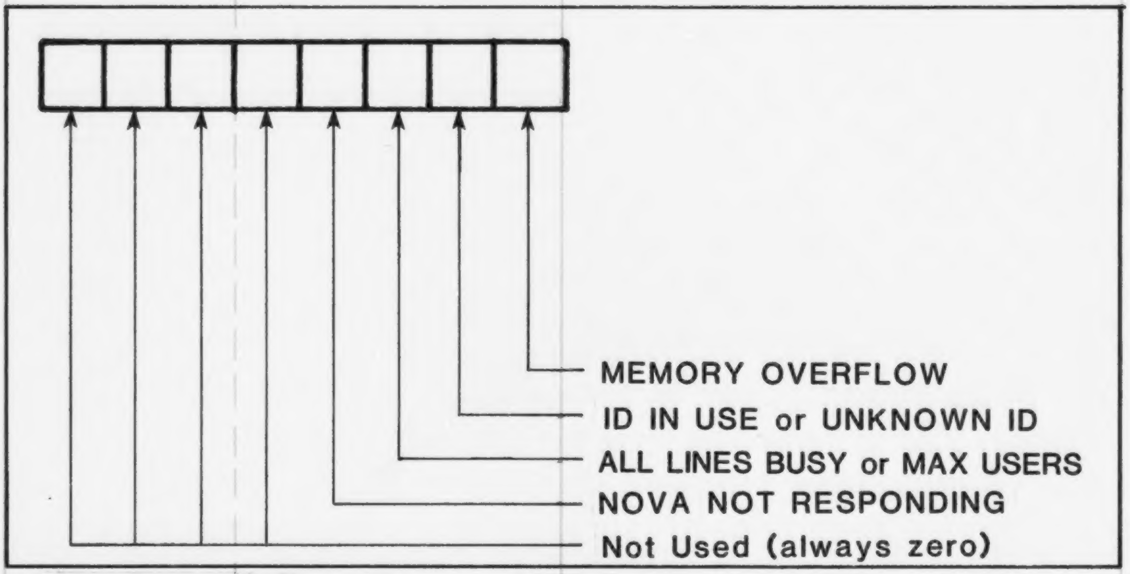

MEMORY OVERFLOW
ID IN USE or UNKNOWN ID
ALL LINES BUSY or MAX USERS
NOVA NOT RESPONDING
Not Used (always zero)

FIGURE 12. SYSTEM STATUS WORD.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA, OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"

PAGE NUMBER
EPA-287 (Cin.)
(4-76)

TYPING GUIDE SHEET

BEGIN FIRST LINE OF TEXT HERE

TOP OF IMAGE AREA

DROPPED HEAD; BEGIN SECTIONS HERE

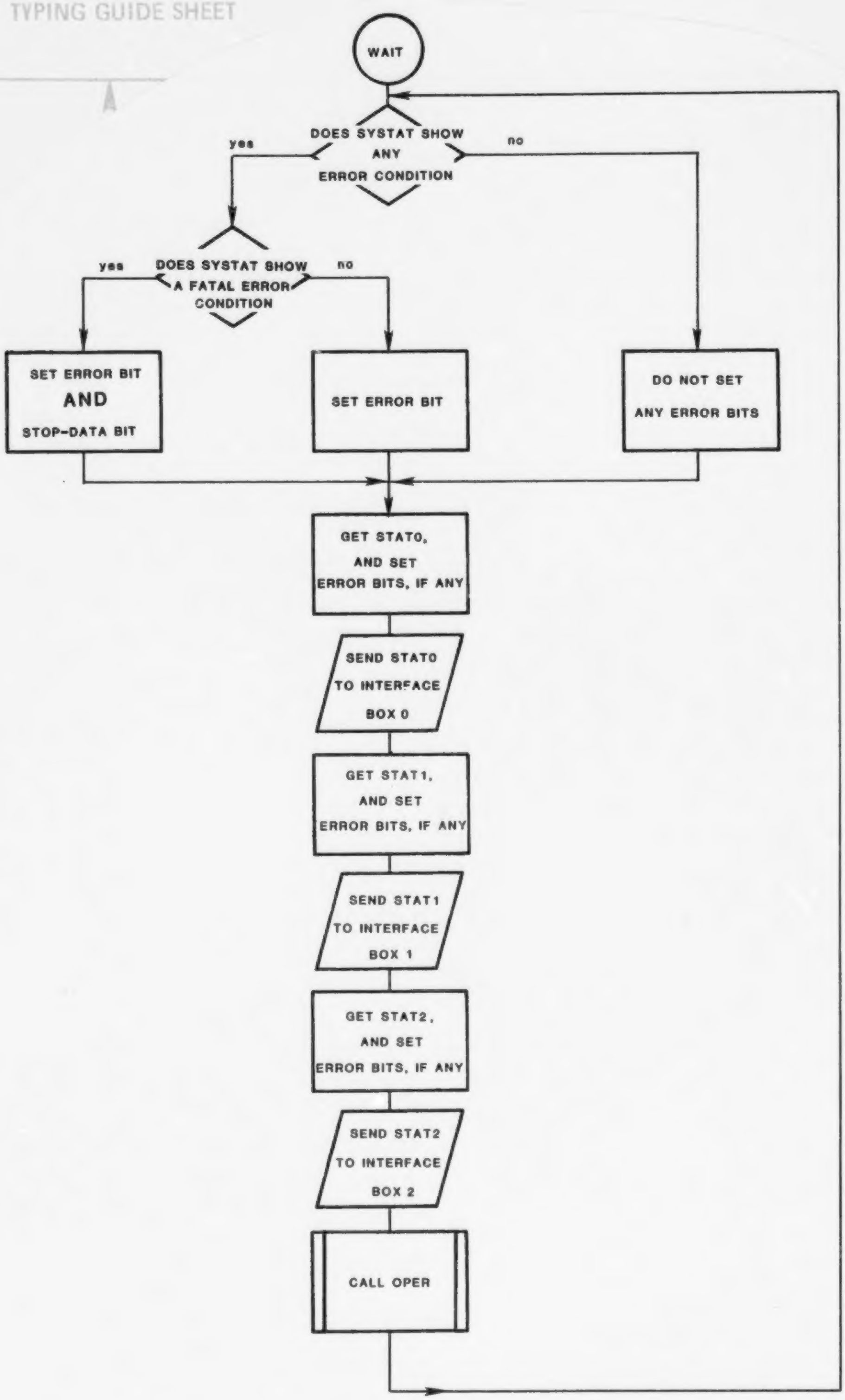

WAIT
DOES SYSTAT SHOW ANY ERROR CONDITION
yes
no
DOES SYSTAT SHOW A FATAL ERROR CONDITION
yes
no
SET ERROR BIT AND STOP-DATA BIT
SET ERROR BIT
DO NOT SET ANY ERROR BITS
GET STAT0, AND SET ERROR BITS, IF ANY
SEND STAT0 TO INTERFACE BOX 0
GET STAT1, AND SET ERROR BITS, IF ANY
SEND STAT1 TO INTERFACE BOX 1
GET STAT2, AND SET ERROR BITS, IF ANY
SEND STAT2 TO INTERFACE BOX 2
CALL OPER

FIGURE 13. WAIT FLOWCHART.

BEGIN LAST LINE OF TEXT

BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

3/8"

PAGE NUMBER

EPA-287 (Cin.)
(4-76)

```
ASM80 :F1:WAIT.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        WAIT     PAGE    1


  LOC  OBJ          LINE        SOURCE STATEMENT

                       1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                       2 ;
                       3 ;          MODULE NAME:  WAIT
                       4 ;
                       5 ;          DESCRIPTION:  THIS MODULE IS A LOOP WHICH HAS LOW PRIORITY.  IT
                       6 ;                        SETS THE ERROR BITS IN THE USER STATUS WORDS AND
                       7 ;                        SENDS THEM TO THE USERS' INTERFACE BOXES.  THIS
                       8 ;                        ROUTINE CALLS THE CONSOLE UPDATE ROUTINE (OPER).
                       9 ;
                      10 ;          MOST RECENT REVISION:  AUGUST 22, 1980
                      11 ;
                      12 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      13 ;
                      14         NAME    WAIT
                      15         PUBLIC  WAIT
                      16         EXTRN   OPER,SYSTAT,STAT
                      17         DSEG
                      18 ;
                      19 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      20 ;
                      21 ;     ROUTINE NAME:  WAIT
                      22 ;
                      23 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      24 ;
  0000 0600           25 WAIT:  MVI     B,00H           ;START WITH NO ERROR BITS IN B-REG
  0002 210000   E     26        LXI     H,SYSTAT        ;LOAD ADDRESS OF SYSTAT INTO H-L-REG
  0005 7E             27        MOV     A,M             ;MOVE SYSTAT INTO A-REG
  0006 FE00           28        CPI     00H             ;ARE THERE ANY SYSTEM ERRORS IN EFFECT?
  0008 CA1600   D     29        JZ      BOX             ;IF NO, JUMP TO BOX
  000B 0640           30        MVI     B,40H           ;SET SEVENTH BIT TO INDICATE NON-FATAL ERROR
  000D E601           31        ANI     01H             ;ELIMINATE ALL BUT FIRST BIT OF SYSTAT
  000F FE01           32        CPI     01H             ;IS FIRST BIT OF SYSTAT 1?
  0011 C21600   D     33        JNZ     BOX             ;IF NO, JUMP TO BOX
  0014 06C0           34        MVI     B,0C0H          ;SET SEVENTH AND EIGHTH BITS TO INDICATE FATAL ERRO
                         R
  0016 3A0000   E     35 BOX:   LDA     STAT            ;LOAD STAT0 INTO A-REG
  0019 B0             36        ORA     B               ;SET ERROR BITS IN STAT0, IF ANY
  001A D310           37        OUT     10H             ;SEND STAT0 TO USER0'S BOX
  001C 3A0001   E     38        LDA     STAT+100H       ;LOAD STAT1 INTO A-REG
  001F B0             39        ORA     B               ;SET ERROR BITS IN STAT1, IF ANY
  0020 D312           40        OUT     12H             ;SEND STAT1 TO USER1'S BOX
  0022 3A0002   E     41        LDA     STAT+200H       ;LOAD STAT2 INTO A-REG
  0025 B0             42        ORA     B               ;SET ERROR BITS IN STAT2, IF ANY
  0026 D314           43        OUT     14H             ;SEND STAT2 TO USER2'S BOX
  0028 CD0000   E     44        CALL    OPER            ;CALL CONSOLE UPDATE PROCEDURE
  002B C30000   D     45        JMP     WAIT            ;JUMP BACK TO WAIT
                      46        END                     ;END OF MODULE

PUBLIC SYMBOLS
WAIT   D 0000

EXTERNAL SYMBOLS
OPER   E 0000    STAT   E 0000    SYSTAT E 0000

USER SYMBOLS
BOX    D 0016    OPER   E 0000    STAT   E 0000    SYSTAT E 0000    WAIT   D 0000
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                    PAGE   1

BOX       29      33      35#
OPER      16      44
STAT      16      35      38      41
SYSTAT    16      26
WAIT      14      15      25#     45

CROSS REFERENCE COMPLETE
```

EPA Form 2350-6 (4-80)
(PREVIOUSLY C/N. EPA FORM 232)
TYPING GUIDE SHEET

7.4 OPER

OPER is the module which updates the system console. It is called as a subroutine at the end of the WAIT module, and it returns to that module when it has completed one screen update. Every time that OPER prints one character on the system console, it first calls the module NOVAST as a subroutine. The result of this arrangement is that NOVAST gets control roughly 120 times per second, which is as fast as the Nova can receive characters from the microprocessor.

As OPER begins, it homes the screen cursor and tabs over to the column for user 0. It then reads the User Status Word, STAT0, from memory. OPER moves the cursor down the column on the console screen, and prints either a blank or an asterisk in each of the positions for "ready," "receiving," "standing by," or "transmitting," depending upon whether the corresponding bit of STAT0 is a 0 or a 1. Similarly, OPER homes the curser, tabs, and updates the column for each of the other two users.

Next, OPER homes the cursor, and tabs it over and down to the position of the system status message. OPER examines the System Status Word, SYSTAT, and prints either "SYSTEM RUNNING NORMALLY" or one of four error messages, as indicated by SYSTAT. The code for the system errors is shown in Figure 12.

Finally, OPER counts the number of memory blocks currently in use. It counts them by examing the first byte of every block and counting those which are not zero. The counting is done in binary coded decimal numbers, which are then converted to ASCII decimal digits and displayed on the console screen.

BEGIN FIRST LINE OF TEXT HERE

TOP OF IMAGE AREA

DROPPED HEAD, BEGIN SECTIONS HERE

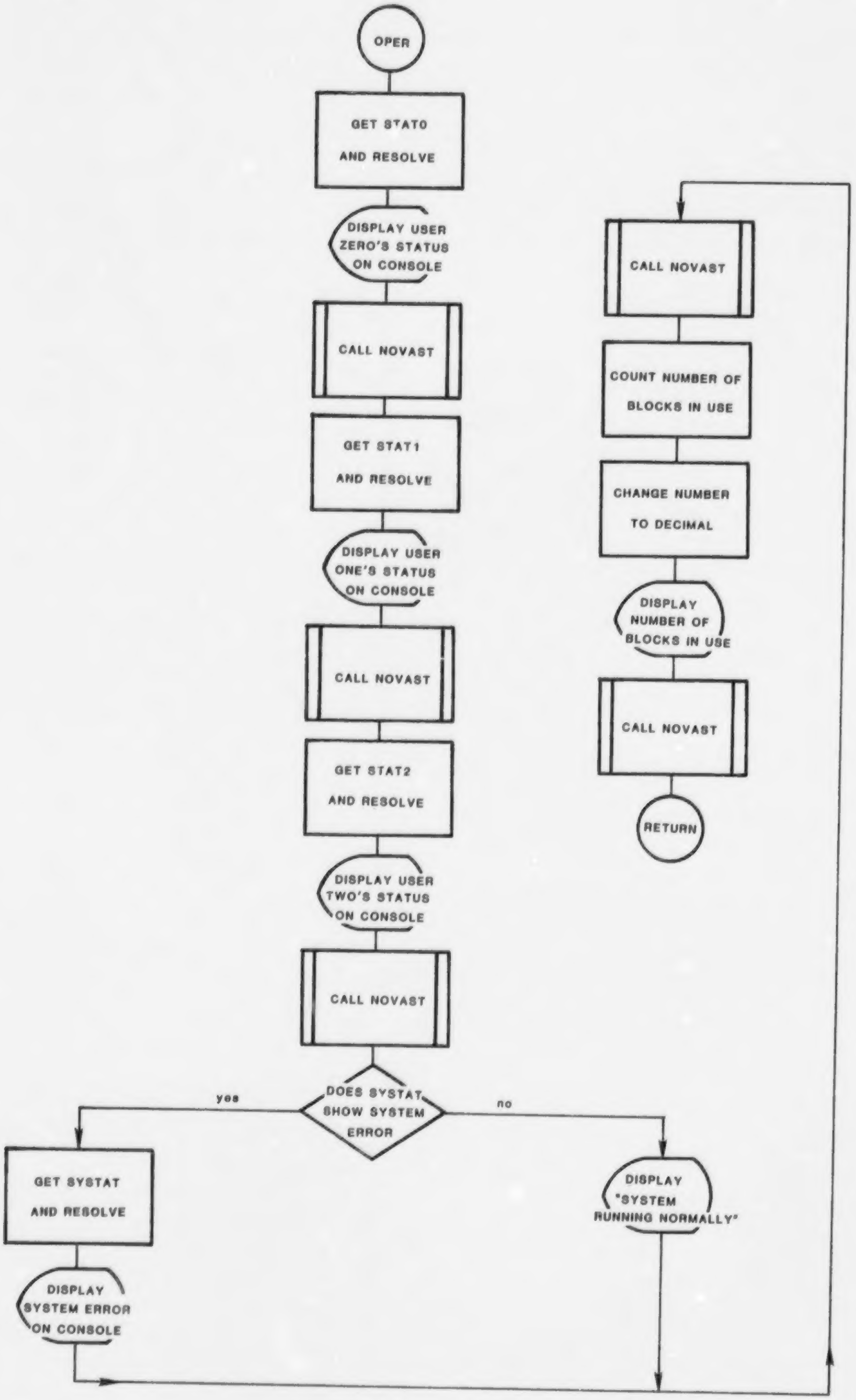

OPER
GET STAT0 AND RESOLVE
DISPLAY USER ZERO'S STATUS ON CONSOLE
CALL NOVAST
GET STAT1 AND RESOLVE
DISPLAY USER ONE'S STATUS ON CONSOLE
CALL NOVAST
GET STAT2 AND RESOLVE
DISPLAY USER TWO'S STATUS ON CONSOLE
CALL NOVAST
DOES SYSTAT SHOW SYSTEM ERROR
yes
no
GET SYSTAT AND RESOLVE
DISPLAY SYSTEM ERROR ON CONSOLE
DISPLAY "SYSTEM RUNNING NORMALLY"
CALL NOVAST
COUNT NUMBER OF BLOCKS IN USE
CHANGE NUMBER TO DECIMAL
DISPLAY NUMBER OF BLOCKS IN USE
CALL NOVAST
RETURN

FIGURE 14. OPER FLOWCHART.

BEGIN LAST LINE OF TEXT

BOTTOM OF IMAGE AREA OUTSIDE DIMENSION FOR TABLES AND ILLUS-TRATIONS

3/8''

PAGE NUMBER

EPA-287 (Cin.)
(4-76)

```
ASM80 :F1:OPER.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        OPER     PAGE    1

  LOC  OBJ         LINE        SOURCE STATEMENT

                      1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      2 ;
                      3 ;         MODULE NAME:  OPER
                      4 ;
                      5 ;         DESCRIPTION:  THIS IS THE CONSOLE UPDATE ROUTINE.  IT HAS THE
                      6 ;                       LOWEST PRIORITY IN THE WHOLE SYSTEM.  IT DISPLAYS
                      7 ;                       ON THE SYSTEM CONSOLE USER AND SYSTEM STATUS.
                      8 ;
                      9 ;         MOST RECENT REVISION:  AUGUST 22, 1980
                     10 ;
                     11 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     12 ;
                     13         NAME    OPER
                     14         PUBLIC  OPER,TAB3,HOME,UPRDY,UPRX,UPSTBY
                     15         EXTRN   STAT,BOTTOM,TOP,CO,BASE,SYSTAT,NOVAST,VECFLG
                     16         DSEG                    ;START IN DATA SEGMENT
                     17 ;
                     18 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     19 ;
                     20 ;     ROUTINE NAME: OPER
                     21 ;     INPUTS: NONE
                     22 ;     OUTPUTS: NONE
                     23 ;     CALLS: MSG,OPMSG,BIT1,BIT2,BIT3,BIT4
                     24 ;
                     25 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     26 ;
0000 016601     D    27 OPER:   LXI     B,HOME          ;HOME
0003 CD3C01     D    28         CALL    OPMSG           ;CURSOR
0006 016301     D    29         LXI     B,TAB0          ;  SET USER
0009 CD3C01     D    30         CALL    OPMSG           ;  0 TAB
000C 210000     E    31         LXI     H,STAT          ;    GET USER 0
000F 5E              32         MOV     E,M             ;    STATUS
0010 CDE800     D    33         CALL    DISPLY          ;RESOLVE & DISPLAY STATUS
0013 016601     D    34         LXI     B,HOME          ;HOME
0016 CD3C01     D    35         CALL    OPMSG           ;CURSOR
0019 016101     D    36         LXI     B,TAB1          ;  SET USER
001C CD3C01     D    37         CALL    OPMSG           ;  1 TAB
001F 210001     E    38         LXI     H,STAT+100H     ;    GET USER 1
0022 5E              39         MOV     E,M             ;    STATUS
0023 CDE800     D    40         CALL    DISPLY          ;RESOLVE & DISPLAY STATUS
0026 016601     D    41         LXI     B,HOME          ;HOME
0029 CD3C01     D    42         CALL    OPMSG           ;CURSOR
002C 015F01     D    43         LXI     B,TAB2          ;  SET USER 2
002F CD3C01     D    44         CALL    OPMSG           ;  TAB
0032 210002     E    45         LXI     H,STAT+200H     ;    GET USER 2
0035 5E              46         MOV     E,M             ;    STATUS
0036 CDE800     D    47         CALL    DISPLY          ;RESOLVE & DISPLAY STATUS
0039 016601     D    48         LXI     B,HOME          ;HOME
003C CD3C01     D    49         CALL    OPMSG           ;CURSOR
003F 015D01     D    50         LXI     B,TAB3          ;  SET SYSTEM
0042 CD3C01     D    51         CALL    OPMSG           ;  STATUS TAB
0045 016901     D    52         LXI     B,UPRDY         ;      SET CURSOR
0048 CD3C01     D    53         CALL    OPMSG           ;      TO CORRECT LINE
004B 3A0000     E    54         LDA     SYSTAT          ;GET SYSTEM STATUS
004E FE00            55         CPI     00H             ;ERROR?
0050 C25900     D    56         JNZ     ERRSET          ;IF YES THEN JUMP
0053 018401     D    57         LXI     B,RUN           ;SET RUN MSG
0056 C37A00     D    58         JMP     COUNT           ;JUMP TO COUNT
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        OPER     PAGE    2


  LOC  OBJ         LINE        SOURCE STATEMENT

                     59 ;
                     60 ;
                     61 ;        *****DISPLAY ERROR MESSAGE*****
                     62 ;
                     63 ;
 0059 1F             64 ERRSET: RAR                          ;ROTATE FIRST BIT INTO CARRY BIT
 005A D26300   D     65         JNC     ERRS2                ;IF FIRST BIT IS NOT 1, JUMP TO ERRS2
 005D 01A901   D     66         LXI     B,ERR1               ;SET MEMORY OVERFLOW ERROR MESSAGE
 0060 C37A00   D     67         JMP     COUNT                ;JUMP TO COUNT
 0063 1F             68 ERRS2:  RAR                          ;ROTATE SECOND BIT INTO CARRY BIT
 0064 D26D00   D     69         JNC     ERRS3                ;IF SECOND BIT IS NOT 1, JUMP TO ERRS3
 0067 01CF01   D     70         LXI     B,ERR2               ;SET ID IN USE ERROR MESSAGE
 006A C37A00   D     71         JMP     COUNT                ;JUMP TO COUNT
 006D 1F             72 ERRS3:  RAR                          ;ROTATE THIRD BIT INTO CARRY BIT
 006E D27700   D     73         JNC     ERRS4                ;IF THIRD BIT IS NOT 1, JUMP TO ERRS4
 0071 01F501   D     74         LXI     B,ERR3               ;SET ALL LINES BUSY ERROR MESSAGE
 0074 C37A00   D     75         JMP     COUNT                ;JUMP TO COUNT
 0077 011B02   D     76 ERRS4:  LXI     B,ERR4               ;SET NOVA NOT RESPONDING ERROR MESSAGE
                     77 ;
                     78 ;
                     79 ;        *****COUNT NUMBER OF BLOCKS IN USE*****
                     80 ;
                     81 ;
 007A CD3C01   D     82 COUNT:  CALL    OPMSG                ;OUTPUT TO CONSOLE
 007D 210000   E     83         LXI     H,BOTTOM             ;GET BOTTOM OF MEM
 0080 010000   E     84         LXI     B,TOP                ;GET TOP OF MEM REGISTER
 0083 0A             85         LDAX    B                    ;GET MSB
 0084 57             86         MOV     D,A                  ;SET IN D REG
 0085 03             87         INX     B                    ;INCREMENT REG
 0086 0A             88         LDAX    B                    ;GET LSB
 0087 5F             89         MOV     E,A                  ;SET IN E REGISTER
 0088 0E00           90         MVI     C,00H                ;ZERO C REGISTER
 008A 7E             91 USE:    MOV     A,M                  ;  BLOCK IN
 008B FE00           92         CPI     00H                  ;  USE?
 008D CA9400   D     93         JZ      NXTBLK               ;IF NOT THEN CONTINUE
 0090 0C             94         INR     C                    ;ADD 1 TO C
 0091 79             95         MOV     A,C                  ;GET COUNT
 0092 27             96         DAA                          ;SET TO BCD
 0093 4F             97         MOV     C,A                  ;RESTORE COUNT
 0094 3E04           98 NXTBLK: MVI     A,04H                ;  INCREMENT MEMORY
 0096 84             99         ADD     H                    ;  POINTER
 0097 67            100         MOV     H,A                  ;RESTORE MEMORY POINTER
 0098 BA            101         CMP     D                    ;AT TOP?
 0099 C28A00   D    102         JNZ     USE                  ;IF NOT THEN JUMP
                    103 ;
                    104 ;
                    105 ;        *****CONVERT NUMBER OF BLOCKS IN USE TO DECIMAL AND DISPLAY*****
                    106 ;
                    107 ;
 009C 79            108 CONVRT: MOV     A,C                  ;GET COUNTER
 009D E60F          109         ANI     0FH                  ;ISOLATE LOW BITS
 009F F6B0          110         ORI     0B0H                 ;SET IN ASCII OFFSET
 00A1 5F            111         MOV     E,A                  ;SET IN E REGISTER
 00A2 79            112         MOV     A,C                  ;GET COUNTER
 00A3 07            113         RLC                          ;  ROTATE
 00A4 07            114         RLC                          ;  MSB
 00A5 07            115         RLC                          ;  AND
 00A6 07            116         RLC                          ;  LSB
 00A7 E60F          117         ANI     0FH                  ;ISOLATE LOW BITS
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        OPER     PAGE    3

  LOC  OBJ          LINE        SOURCE STATEMENT

 00A9 F6B0           118         ORI     0B0H           ;SET ASCII OFFSET
 00AB 57             119         MOV     D,A            ;SET IN D REGISTER
 00AC D5             120         PUSH    D              ;SAVE REGISTER
 00AD 016601    D    121         LXI     B,HOME         ;HOME
 00B0 CD3C01    D    122         CALL    OPMSG          ;CURSOR
 00B3 015D01    D    123         LXI     B,TAB3         ;SET SYSTEM TAB
 00B6 CD3C01    D    124         CALL    OPMSG          ;ON CONSOLE
 00B9 016901    D    125         LXI     B,UPRDY        ;  SET CURSOR
 00BC CD3C01    D    126         CALL    OPMSG          ;  TO CORRECT
 00BF 017401    D    127         LXI     B,UPRX         ;  POSITION
 00C2 CD3C01    D    128         CALL    OPMSG          ;  ********
 00C5 014102    D    129         LXI     B,BLOCKS       ;  HEADER
 00C8 CD3C01    D    130         CALL    OPMSG          ;  FOR BLOCKS
 00CB D1             131         POP     D              ;RESTORE REGISTER
 00CC 4A             132         MOV     C,D            ;GET HIGH BYTE
 00CD 3A0000    E    133         LDA     VECFLG         ;LOAD VECFLG INTO A-REG
 00D0 FE00           134         CPI     00H            ;IS VECFLG SET TO ZERO (LIGHT VECTOR MODE)?
 00D2 C2E200    D    135         JNZ     CON1           ;IF NO, JUMP TO CON1
 00D5 CD0000    E    136         CALL    CO             ;OUT TO CONSOLE
 00D8 4B             137         MOV     C,E            ;GET LOW BYTE
 00D9 CD0000    E    138         CALL    CO             ;OUT TO CONSOLE
 00DC 015402    D    139         LXI     B,SPACE        ;LOAD ADDRESS OF SPACE INTO B-C-REG
 00DF CD3C01    D    140         CALL    OPMSG          ;PRINT SPACES
 00E2 210000    E    141 CON1:   LXI     H,VECFLG       ;LOAD ADDRESS OF VECFLG INTO H-L-REG
 00E5 3600           142         MVI     M,00H          ;SET VECFLG TO ZERO (LIGHT VECTOR MODE)
 00E7 C9             143         RET                    ;RETURN
                     144 ;
                     145 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     146 ;
                     147 ;      ROUTINE NAME:  DISPLAY
                     148 ;      DESCRIPTION:   THIS SUBROUTINE PLACES ASTERISKS IN THE APPROPRIATE
                     149 ;                     COLUMNS TO INDICATE READY, RECEIVING, STANDBY, AND/OR
                     150 ;                     TRANSMITTING.
                     151 ;
                     152 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     153 ;
 00E8 016901    D    154 DISPLY: LXI     B,UPRDY        ;  SET CURSOR
 00EB CD3C01    D    155         CALL    OPMSG          ;  TO CORRECT POSITION
 00EE CD0D01    D    156         CALL    BIT1           ;TEST BIT 1
 00F1 017401    D    157         LXI     B,UPRX         ;  SET CURSOR TO
 00F4 CD3C01    D    158         CALL    OPMSG          ;  CORRECT POSITION
 00F7 CD1601    D    159         CALL    BIT2           ;TEST BIT 2
 00FA 017801    D    160         LXI     B,UPSTBY       ;  SET CURSOR TO
 00FD CD3C01    D    161         CALL    OPMSG          ;  CORRECT POSITION
 0100 CD1F01    D    162         CALL    BIT3           ;TEST BIT 3
 0103 017C01    D    163         LXI     B,UPTX         ;  SET CURSOR TO
 0106 CD3C01    D    164         CALL    OPMSG          ;  CORRECT POSITION
 0109 CD2801    D    165         CALL    BIT4           ;TEST BIT 4
 010C C9             166         RET                    ;RETURN
 010D 7B             167 BIT1:   MOV     A,E            ;COPY REGISTER
 010E E601           168         ANI     01H            ;ISOLATE BIT
 0110 CA3501    D    169         JZ      DISIDL         ;IF ZERO JUMP
 0113 C32E01    D    170         JMP     DISACT         ;JUMP
 0116 7B             171 BIT2:   MOV     A,E            ;COPY REGISTER
 0117 E602           172         ANI     02H            ;ISOLATE BIT
 0119 CA3501    D    173         JZ      DISIDL         ;IF ZERO JUMP
 011C C32E01    D    174         JMP     DISACT         ;JUMP
 011F 7B             175 BIT3:   MOV     A,E            ;COPY REGISTER
 0120 E604           176         ANI     04H            ;ISOLATE BIT
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        OPER     PAGE    4

  LOC  OBJ         LINE       SOURCE STATEMENT

  0122 CA3501   D   177           JZ      DISIDL          ;IF ZERO JUMP
  0125 C32E01   D   178           JMP     DISACT          ;JUMP
  0128 7B           179 BIT4:     MOV     A,E             ;COPY REGISTER
  0129 E608         180           ANI     08H             ;ISOLATE BIT
  012B CA3501   D   181           JZ      DISIDL          ;IF ZERO JUMP
  012E 018001   D   182 DISACT:   LXI     B,ACTIVE        ;LOAD STAR
  0131 CD3C01   D   183           CALL    OPMSG           ;OUTPUT TO CONSOLE
  0134 C9           184           RET                     ;RETURN
  0135 018201   D   185 DISIDL:   LXI     B,IDLE          ;LOAD BLANK
  0138 CD3C01   D   186           CALL    OPMSG           ;OUTPUT TO CONSOLE
  013B C9           187           RET                     ;RETURN
                    188 ;
                    189 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:;
                    190 ;
                    191 ;       ROUTINE NAME: OPMSG
                    192 ;       INPUTS: BC REG PAIR
                    193 ;       OUTPUTS: NONE
                    194 ;       CALLS: CO,NOVAST
                    195 ;       DESCRIPTION: THIS ROUTINE IS A SPECIAL CONSOLE OUTPUT ROUTINE THAT CALLS
                    196 ;                    THE NOVA DUMP STATUS EACH TIME THERE IS TO BE AN OUTPUT ON
                    197 ;                    THE CONSOLE SCREEN.  IT IS USED ONLY IN THIS MODULE.
                    198 ;
                    199 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    200 ;
  013C CD0000   E   201 OPMSG:    CALL    NOVAST          ;CALL DUMP STATUS CHECK ROUTINE
  013F 0A           202           LDAX    B               ;GET LETTER
  0140 C5           203           PUSH    B               ;SAVE REGISTER
  0141 D5           204           PUSH    D               ;SAVE REGISTER
  0142 E5           205           PUSH    H               ;SAVE REGISTER
  0143 F5           206           PUSH    PSW             ;SAVE REGISTER
  0144 F680         207           ORI     80H             ;SET PARITY
  0146 4F           208           MOV     C,A             ;  SEND CHARACTER OUT
  0147 3A0000   E   209           LDA     VECFLG          ;LOAD VECFLG INTO A-REG
  014A FE00         210           CPI     00H             ;IS VECFLG SET TO ZERO (LIGHT VECTOR MODE)?
  014C C25201   D   211           JNZ     OPM1            ;IF NO, JUMP TO OPM1
  014F CD0000   E   212           CALL    CO              ;  TO CONSOLE
  0152 F1           213 OPM1:     POP     PSW             ;RESET STACK
  0153 E1           214           POP     H               ;RESET STACK
  0154 D1           215           POP     D               ;RESET STACK
  0155 C1           216           POP     B               ;RESET STACK
  0156 FE00         217           CPI     00H             ;  ON <NULL CHARACTER>
  0158 C8           218           RZ                      ;  THEN RETURN
  0159 03           219           INX     B               ;NEXT LETTER
  015A C33C01   D   220           JMP     OPMSG           ;CONTINUE
                    221 ;
                    222 ;
                    223 ;         *****CONSTANTS*****
                    224 ;
                    225 ;
  015D 1B           226 TAB3:     DB      1BH,49H
  015E 49
  015F 1B           227 TAB2:     DB      1BH,49H
  0160 49
  0161 1B           228 TAB1:     DB      1BH,49H
  0162 49
  0163 1B           229 TAB0:     DB      1BH,49H,00H
  0164 49
  0165 00
  0166 1B           230 HOME:     DB      1BH,48H,00H
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

 0167 48
 0168 00
 0169 0A            231 UPRDY:  DB      0AH,0AH,0AH,0AH,0AH,0AH,0AH,0AH
 016A 0A
 016B 0A
 016C 0A
 016D 0A
 016E 0A
 016F 0A
 0170 0A
 0171 0A            232         DB      0AH,08H,00H
 0172 08
 0173 00
 0174 0A            233 UPRX:   DB      0AH,0AH,08H,00H
 0175 0A
 0176 08
 0177 00
 0178 0A            234 UPSTBY: DB      0AH,0AH,08H,00H
 0179 0A
 017A 08
 017B 00
 017C 0A            235 UPTX:   DB      0AH,0AH,08H,00H
 017D 0A
 017E 08
 017F 00
 0180 2A            236 ACTIVE: DB      2AH,00H
 0181 00
 0182 20            237 IDLE:   DB      20H,00H
 0183 00
 0184 0A            238 RUN:    DB      0AH,'SYSTEM RUNNING NORMALLY          ',00H
 0185 53595354
 0189 454D2052
 018D 554E4E49
 0191 4E47204E
 0195 4F524D41
 0199 4C4C5920
 019D 20202020
 01A1 20202020
 01A5 202020
 01A8 00
 01A9 0A            239 ERR1:   DB      0AH,'ERROR:  MEMORY OVERFLOW          ',07H,00H
 01AA 4552524F
 01AE 523A2020
 01B2 4D454D4F
 01B6 5259204F
 01BA 56455246
 01BE 4C4F5720
 01C2 20202020
 01C6 20202020
 01CA 202020
 01CD 07
 01CE 00
 01CF 0A            240 ERR2:   DB      0AH,'ERROR:  ID IN USE OR UNKNOWN ID  ',07H,00H
 01D0 4552524F
 01D4 523A2020
 01D8 49442049
 01DC 4E205553
 01E0 45204F52
 01E4 20554E4B
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        OPER     PAGE    6

  LOC  OBJ         LINE        SOURCE STATEMENT

  01E8 4E4F574E
  01EC 20494420
  01F0 202020
  01F3 07
  01F4 00
  01F5 0A           241 ERR3:   DB      0AH,'ERROR:  ALL LINES BUSY OR MAX USERS',07H,00H
  01F6 4552524F
  01FA 523A2020
  01FE 414C4C20
  0202 4C494E45
  0206 53204255
  020A 5359204F
  020E 52204D41
  0212 58205553
  0216 455253
  0219 07
  021A 00
  021B 0A           242 ERR4:   DB      0AH,'ERROR:  NOVA NOT RESPONDING        ',07H,00H
  021C 4552524F
  0220 523A2020
  0224 4E4F5641
  0228 204E4F54
  022C 20524553
  0230 504F4E44
  0234 494E4720
  0238 20202020
  023C 202020
  023F 07
  0240 00
  0241 20           243 BLOCKS: DB      20H,0AH,'BLOCKS IN USE:  ',00H
  0242 0A
  0243 424C4F43
  0247 4B532049
  024B 4E205553
  024F 453A2020
  0253 00
  0254 1B           244 SPACE:  DB      1BH,4BH,00H
  0255 4B
  0256 00
                    245         END                     ;END OF MODULE

PUBLIC SYMBOLS
HOME   D 0166    OPER   D 0000    TAB3   D 015D    UPRDY  D 0169    UPRX   D 0174    UPSTBY D 0178

EXTERNAL SYMBOLS
BASE   E 0000    BOTTOM E 0000    CO     E 0000    NOVAST E 0000    STAT   E 0000    SYSTAT E 0000
TOP    E 0000    VECFLG E 0000

USER SYMBOLS
ACTIVE D 0180    BASE   E 0000    BIT1   D 010D    BIT2   D 0116    BIT3   D 011F    BIT4   D 0128
BLOCKS D 0241    BOTTOM E 0000    CO     E 0000    CON1   D 00E2    CONVRT D 009C    COUNT  D 007A
DISACT D 012E    DISIDL D 0135    DISPLY D 00E8    ERR1   D 01A9    ERR2   D 01CF    ERR3   D 01F5
ERR4   D 021B    ERRS2  D 0063    ERRS3  D 006D    ERRS4  D 0077    ERRSET D 0059    HOME   D 0166
IDLE   D 0182    NOVAST E 0000    NXTBLK D 0094    OPER   D 0000    OPM1   D 0152    OPMSG  D 013C
RUN    D 0184    SPACE  D 0254    STAT   E 0000    SYSTAT E 0000    TAB0   D 0163    TAB1   D 0161
TAB2   D 015F    TAB3   D 015D    TOP    E 0000    UPRDY  D 0169    UPRX   D 0174    UPSTBY D 0178
UPTX   D 017C    USE    D 008A    VECFLG E 0000
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                   PAGE   1

ACTIVE  182   236#
BASE     15
BIT1    156   167#
BIT2    159   171#
BIT3    162   175#
BIT4    165   179#
BLOCKS  129   243#
BOTTOM   15    83
CO       15   136   138   212
CON1    135   141#
CONVRT  108#
COUNT    58    67    71    75    82#
DISACT  170   174   178   182#
DISIDL  169   173   177   181   185#
DISPLY   33    40    47   154#
ERR1     66   239#
ERR2     70   240#
ERR3     74   241#
ERR4     76   242#
ERRS2    65    68#
ERRS3    69    72#
ERRS4    73    76#
ERRSET   56    64#
HOME     14    27    34    41    48   121   230#
IDLE    185   237#
NOVAST   15   201
NXTBLK   93    98#
OPER     13    14    27#
OPM1    211   213#
OPMSG    28    30    35    37    42    44    49    51    53    82   122   124   126   128
        130   140   155   158   161   164   183   186   201#  220
RUN      57   238#
SPACE   139   244#
STAT     15    31    38    45
SYSTAT   15    54
TAB0     29   229#
TAB1     36   228#
TAB2     43   227#
TAB3     14    50   123   226#
TOP      15    84
UPRDY    14    52   125   154   231#
UPRX     14   127   157   233#
UPSTBY   14   160   234#
UPTX    163   235#
USE      91#  102
VECFLG   15   133   141   209

CROSS REFERENCE COMPLETE
```

7.5 USER

USER is the module which takes a character arriving from an intelligent instrument, determines if it is part of report, and if it is, stores it in memory. There are actually three separate routines called USER0, USER1, and USER2 in this module, but they all perform the same functions for the three different users. Module TB534 directs control of the computer to the proper routine. USER is called as a subroutine from TB534, and it returns control of the computer to that module when it has completed its tasks.

When USER first gets control, it reads from the UART the character which caused the interrupt. It then checks the proper User Status Word to see whether the system is currently in the process of receiving a report from the instrument, or if it is waiting for a new report to begin. If the system is waiting for a new report to begin, it checks the incoming character to see if it is one of the five starting characters. When the first of the five starting characters is recognized, the system reserves a block of memory and begins placing characters into it. If the next four incoming characters do not complete the proper sequence of five starting characters, the system frees the block of memory. Conversely, when the system recognizes that the entire string of five starting characters has been received, it changes the User Status Word from "ready" to "receiving" and continues to fill the memory block.

To store a character, USER first checks whether a block of memory has been allocated yet for the report. If not, USER calls the subroutine FRESH which locates an available block of memory. (FRESH is described separately below.) A block of memory is a one-kilobyte section of RAM. The first block extends from address 4000_H to $43FF_H$, the second from 4400_H to $47FF_H$, the third from 4800_H to $4BFF_H$, and so forth. There are 16, 32, or 48 blocks available, depending on the number of memory expansion boards in the host computer. The USER starts a new block of memory, it initializes it by writing into it the data shown in Figure 15.

If a block of memory has already been assigned for the report, USER checks to see if the block is full. If it is, USER calls the subroutine FRESH to locate an available block, initializes that block, and writes the most significant byte of the address of the new block into the last byte of the old block. (The least significant byte does not have to be stored since it is always zero.)

RELATIVE ADDRESS	DATA
0000	User Code: 01=User 0, 02=User 1, 03=User 2, 00=block not in use
0001	Report Status Flag: 00=Incomplete, FF=Complete
0002	Least significant byte of number of bytes used in this block
0003	Most significant byte of number of bytes used in this block
0004	Report Sequence Number
0005	Character of report
0006	Character of report
.	.
.	.
.	.
03FD	Character of report
03FE	Character of report
03FF	Most significant byte of address of next block linked, if any

Figure 15. Memory Block Configuration.

USER then checks again whether it is in the process of receiving a report or searching for the beginning of one. If it is still checking for the five starting characters, it simply stores the latest character, updates the pointers for where to store the next character, and returns control to TB534. If USER has already detected the five starting characters, it must check each subsequent character to see if it is part of the string of five stopping characters. When the first stopping character is detected, USER starts counting the stopping characters. Naturally, it resets its counter to zero if it discovers that the characters it counted were not part of the complete string of stopping characters. When USER has detected the whole string of stopping characters, it continues to store incoming characters until it detects a carriage return. This ensures that the system saves the entire last line of the report. When the carriage return ending the report is found, USER changes the User Status Word to indicate that the system is "ready" to receive another report and "standing by" with a completed report. It sets the Report Status Flag in the second byte of the first block of the report to indicate that the report is complete, and it increments the standby counter, STBYn, so that NOVAST will know that there is a report to be sent to the Nova.

All the activity in USER is fragmented. USER handles only one character each time it gets control of the computer.

Subroutine FRESH

The subroutine FRESH performs a very simple task with one gigantic complication. FRESH is called to find an available block of memory. Normally, it simply starts at the first block of memory and checks the first byte. If the first byte of the block is not 00, FRESH knows that the block is in use, and it goes on to test the next block. When it finds a free block, it places the address of that block into the B-C register pair, and it returns to the invoking routine.

The obvious complication is the memory overflow situation which occurs when an incoming character causes a call to FRESH, but there are no free blocks left. In this case, FRESH takes over the duties of the WAIT loop with special conditions until the memory overflow situation is resolved. Thus, when FRESH cannot find a free block of memory, it immediately sends a USER Status Word with the "error" and "stop data" bits set to every interface box. Circuitry in the interface boxes stops the intelligent instruments from transmitting, if they are intelligent enough to respond to this feature. FRESH then masks the interrupts from the intelligent instruments, resets the programmable interrupt controllers, sets the memory overflow bit in the System Status Word, enables the one remaining unmasked interrupt associated with the Nova, and calls the OPER subroutine. OPER calls NOVAST, which sends characters to the Nova. Enabling the interrupt allows the NOVA to echo those characters back to the Microprocessor Buffer System. (The interrupt must be enabled because it was disabled by the interrupt which caused the overflow.) FRESH continues to cycle in the manner of the WAIT loop, repeatedly updating the console and the interface box lights, but also searching for a free block or memory. When NOVAST has sent a report to the Nova, it frees the blocks of memory occupied by that report. Then FRESH disables all interrupts, unmasks the instrument interrupts, resets the memory overflow bit in the System Status Word, puts the address of the free memory block into the B-C register pair, and returns to USER. (The interrupts are re-enabled by TB534 just before that routine restores control to the procedure which was interrupted by the character which caused the overflow.)

There is one other obscure aspect to this procedure. The incoming character which causes a memory overflow may interrupt OPER part way through a console update, and FRESH calls OPER in a memory overflow situation. OPER prints many escape sequences to the console for cursor control purposes. If the last character sent to the console before the interrupt is part of an escape sequence, and then FRESH starts OPER over from the beginning, the result would be a garbled escape sequence with random consequences. Therefore, before FRESH calls OPER for the first time, it sends a bell character to the console. This creats an invalid escape sequence which is ignored by the console. Similarly, when FRESH reliquishes control, OPER would resume from the point where it was interrupted, but the cursor would not still be where OPER left it at the time of the interrupt. Therefore, FRESH sets a flag called VECFLG which causes OPER to print nothing but null characters from the time that it regains control until it finishes one console update.

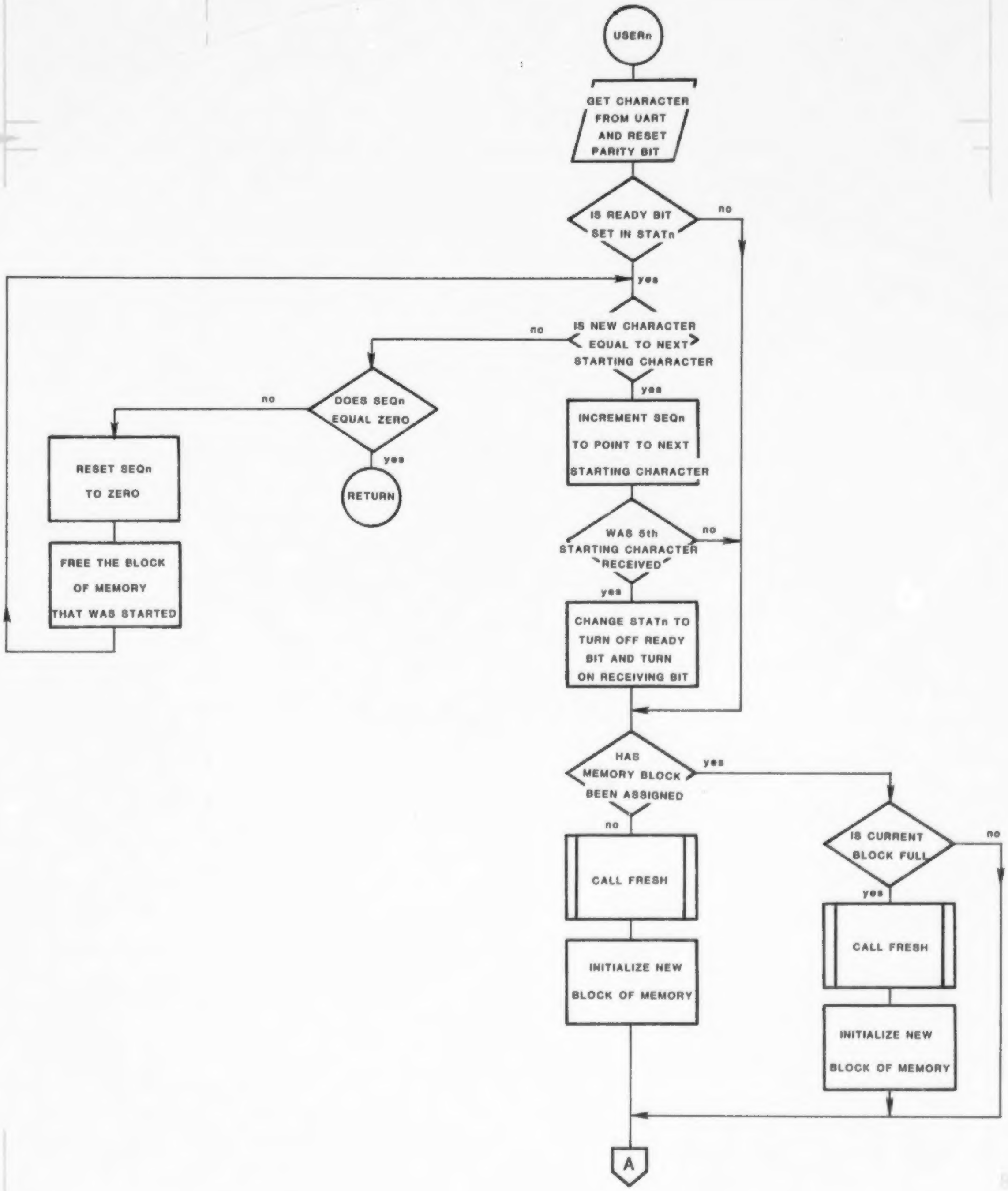
BEGIN FIRST LINE OF TEXT HERE

TYPING GUIDE SHEET

CENTER OF PAGE

TOP OF IMAGE AREA

DROPPED HEAD BEGIN SECTIONS HERE


USERn
GET CHARACTER FROM UART AND RESET PARITY BIT
IS READY BIT SET IN STATn
no
yes
IS NEW CHARACTER EQUAL TO NEXT STARTING CHARACTER
no
yes
DOES SEQn EQUAL ZERO
no
yes
RETURN
RESET SEQn TO ZERO
FREE THE BLOCK OF MEMORY THAT WAS STARTED
INCREMENT SEQn TO POINT TO NEXT STARTING CHARACTER
WAS 5th STARTING CHARACTER RECEIVED
no
yes
CHANGE STATn TO TURN OFF READY BIT AND TURN ON RECEIVING BIT
HAS MEMORY BLOCK BEEN ASSIGNED
yes
no
CALL FRESH
INITIALIZE NEW BLOCK OF MEMORY
IS CURRENT BLOCK FULL
no
yes
CALL FRESH
INITIALIZE NEW BLOCK OF MEMORY
A

BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

BEGIN LAST LINE OF TEXT

FIGURE 16. USER FLOWCHART (PART A).

3/8"

PAGE NUMBER

EPA-287 (Cin.)
(4-70)

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD; BEGIN SECTIONS HERE
BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

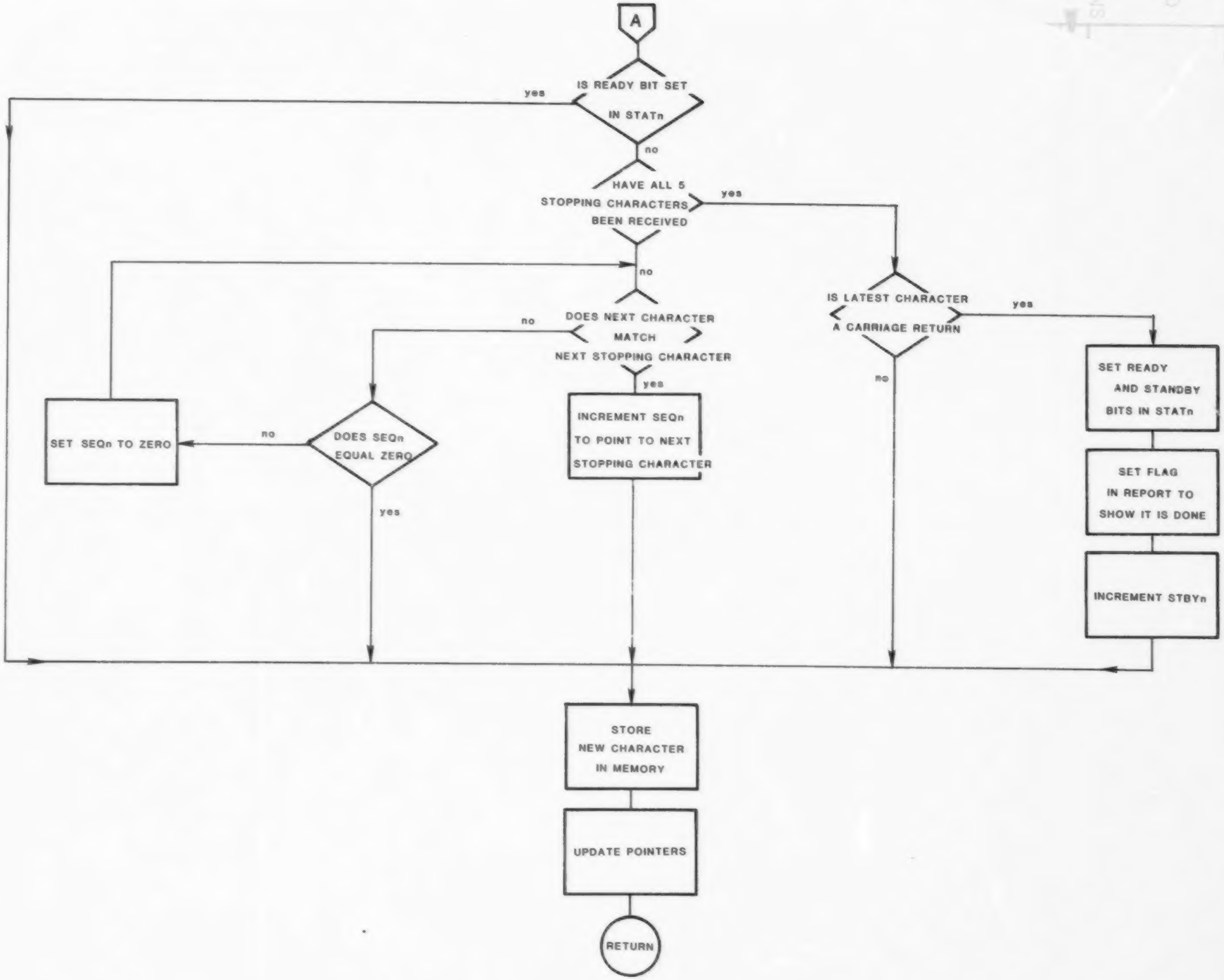

A
IS READY BIT SET IN STATn
yes
no
HAVE ALL 5 STOPPING CHARACTERS BEEN RECEIVED
yes
no
IS LATEST CHARACTER A CARRIAGE RETURN
yes
no
SET READY AND STANDBY BITS IN STATn
SET FLAG IN REPORT TO SHOW IT IS DONE
INCREMENT STBYn
DOES NEXT CHARACTER MATCH NEXT STOPPING CHARACTER
no
yes
INCREMENT SEQn TO POINT TO NEXT STOPPING CHARACTER
DOES SEQn EQUAL ZERO
no
yes
SET SEQn TO ZERO
STORE NEW CHARACTER IN MEMORY
UPDATE POINTERS
RETURN

FIGURE 17. USER FLOWCHART (PART B).

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
TOP OF IMAGE AREA
DROPPED HEAD, BEGIN SECTIONS HERE

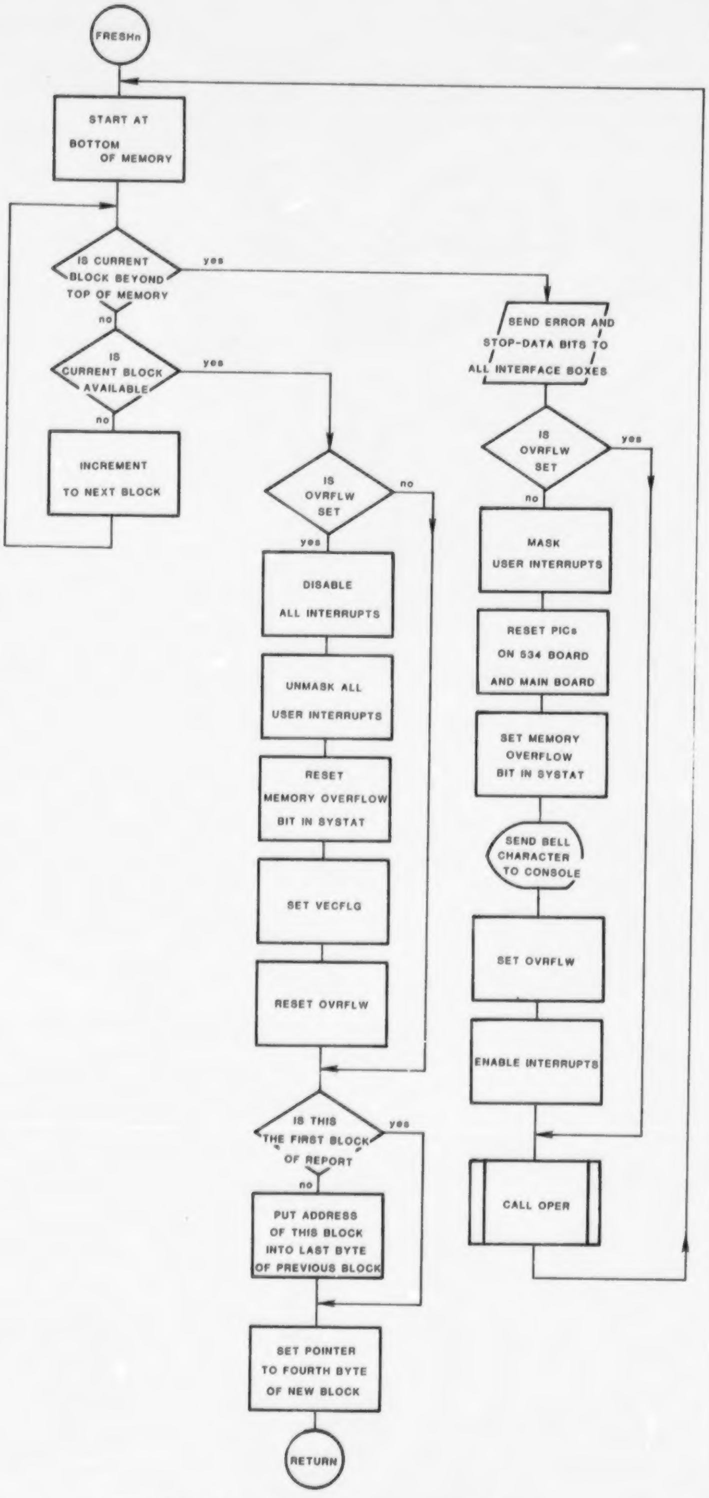

FRESHn
START AT BOTTOM OF MEMORY
IS CURRENT BLOCK BEYOND TOP OF MEMORY
yes
no
SEND ERROR AND STOP-DATA BITS TO ALL INTERFACE BOXES
IS CURRENT BLOCK AVAILABLE
yes
no
INCREMENT TO NEXT BLOCK
IS OVRFLW SET
no
yes
IS OVRFLW SET
yes
no
MASK USER INTERRUPTS
DISABLE ALL INTERRUPTS
RESET PICs ON 534 BOARD AND MAIN BOARD
UNMASK ALL USER INTERRUPTS
SET MEMORY OVERFLOW BIT IN SYSTAT
RESET MEMORY OVERFLOW BIT IN SYSTAT
SEND BELL CHARACTER TO CONSOLE
SET VECFLG
SET OVRFLW
RESET OVRFLW
ENABLE INTERRUPTS
IS THIS THE FIRST BLOCK OF REPORT
yes
no
CALL OPER
PUT ADDRESS OF THIS BLOCK INTO LAST BYTE OF PREVIOUS BLOCK
SET POINTER TO FOURTH BYTE OF NEW BLOCK
RETURN

FIGURE 18. FRESH FLOWCHART.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA, OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

```
ASM80 :F1:USER.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE    1

  LOC  OBJ         LINE        SOURCE STATEMENT

                      1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      2 ;
                      3 ;           MODULE NAME:  USER
                      4 ;
                      5 ;           DESCRIPTION:  THIS MODULE IS ACTIVATED WHEN A CHARACTER
                      6 ;                         ARRIVING FROM ONE OF THE INTELLIGENT
                      7 ;                         INSTRUMENTS CAUSES AN INTERRUPT.  THE CHARACTER
                      8 ;                         IS STORED IN MEMORY BY THIS MODULE.
                      9 ;
                     10 ;           MOST RECENT REVISION:  AUGUST 26, 1980
                     11 ;
                     12 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     13 ;
                     14         NAME    USER
                     15         PUBLIC  USER0,USER1,USER2
                     16         EXTRN   EOI534,EOIM,CO
                     17         EXTRN   TTYBUF,STAT,SYSTAT,BOTTOM,TOP,OPER,BLK0,BLK1,BLK2,SEQ0,SEQ1,SEQ2
                     18         EXTRN   STBY0,STBY1,STBY2,OVRFLW,VECFLG,ZERO1
                     19         DSEG                    ;START IN DATA SEGMENT
                     20 ;
                     21 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     22 ;
                     23 ;       ROUTINE NAME: USER0
                     24 ;       INPUTS: ASCII CHARACTER IN D REGISTER
                     25 ;       OUTPUTS: NONE
                     26 ;       CALLS: OPER,ZERO1,FRESH,RESET
                     27 ;
                     28 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     29 ;
  0000 DB10          30 USER0:  IN      10H             ;GET CHARACTER FROM PORT 0
  0002 E67F          31         ANI     7FH             ;RESET PARITY BIT
  0004 5F            32         MOV     E,A             ;STORE INPUT BYTE
  0005 3A0000    E   33         LDA     STAT            ;GET USER STATUS
  0008 E601          34         ANI     01H             ;ISOLATE BIT 1
  000A FE01          35         CPI     01H             ;IS USER READY?
  000C C24500    D   36         JNZ     BLOCK0          ;IF SO THEN CONTINUE
  000F 3A0000    E   37 RECHK0: LDA     SEQ0            ;GET ROUTINE STATUS
  0012 210000    E   38         LXI     H,TTYBUF        ;GET START SEQUENCE
  0015 8D            39         ADC     L               ;SET POINTER
  0016 6F            40         MOV     L,A             ;RESTORE POINTER
  0017 7E            41         MOV     A,M             ;PUT LETTER IN ACC
  0018 210000    E   42         LXI     H,SEQ0          ;GET SEQUENCE COUNTER
  001B BB            43         CMP     E               ;COMPARE WITH INPUT BYTE
  001C CA3500    D   44         JZ      CONT0           ;IF DIFFERENT THEN JUMP
  001F 7E            45         MOV     A,M             ;  COUNTER
  0020 FE00          46         CPI     00H             ;AT ZERO?
  0022 C8            47         RZ                      ;RETURN
  0023 CD5D03    D   48         CALL    RESET           ;IF NO THEN RESET COUNTER
  0026 210000    E   49         LXI     H,BLK0          ;GET BLOCK COUNTER
  0029 46            50         MOV     B,M             ;GET VALUE
  002A AF            51         XRA     A               ;ZERO OUT ACCUMULATER
  002B 4F            52         MOV     C,A             ;ZERO OUT C REGISTER
  002C 02            53         STAX    B               ;ZERO OUT BYTE 1 IN BLOCK
  002D 77            54         MOV     M,A             ;ZERO OUT BLOCK COUNTER
  002E 23            55         INX     H               ;GET BLOCK POINTER LSB
  002F 77            56         MOV     M,A             ;ZERO OUT BLOCK POINTER
  0030 23            57         INX     H               ;GET BLOCK POINTER MSB
  0031 77            58         MOV     M,A             ;ZERO OUT BLOCK POINTER
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE    2

 LOC  OBJ          LINE        SOURCE STATEMENT

 0032 C30F00   D     59             JMP     RECHK0             ;JUMP TO RECHECK CHARACTER
 0035 7E             60 CONT0:      MOV     A,M                ;  COUNTER
 0036 3C             61             INR     A                  ;  COUNTER
 0037 77             62             MOV     M,A                ;SET IN MEMORY
 0038 FE05           63             CPI     05H                ;CHECK FOR FULL COUNT
 003A C24500   D     64             JNZ     BLOCK0             ;IF NOT FULL THEN JUMP
 003D 210000   E     65             LXI     H,STAT             ;  BYTE TO RCVG
 0040 34             66             INR     M                  ;SET TO RCVG
 0041 AF             67             XRA     A                  ;ZERO ACCUMULATER
 0042 320000   E     68             STA     SEQ0               ;SET SEQUENCE COUNTER TO ZERO
 0045 3A0000   E     69 BLOCK0:     LDA     BLK0               ;GET BLOCK COUNTER
 0048 FE00           70             CPI     00H                ;BLOCK ASSIGNED?
 004A C25F00   D     71             JNZ     POINT0             ;IF YES THEN JUMP
 004D CDEC00   D     72             CALL    FRESH0             ;GET FRESH BLOCK
 0050 210100   E     73             LXI     H,STBY0+01H        ;GET REGISTER
 0053 70             74             MOV     M,B                ;  SET ADDRESS
 0054 23             75             INX     H                  ;  IN B
 0055 71             76             MOV     M,C                ;  REGISTER PAIR
 0056 AF             77             XRA     A                  ;ZERO ACCUMULATER
 0057 02             78             STAX    B                  ;SET BYTE 1
 0058 03             79             INX     B                  ;INCREMENT BLOCK
 0059 02             80             STAX    B                  ;SET BYTE 2
 005A 03             81             INX     B                  ;INCREMENT BLOCK
 005B 02             82             STAX    B                  ;SET BYTE 3
 005C C37800   D     83             JMP     STOP0              ;JUMP
 005F 210200   E     84 POINT0:     LXI     H,SEQ0+2                 ;  GET BLOCK
 0062 7E             85             MOV     A,M                ;  POINTER LSB
 0063 FEFF           86             CPI     0FFH               ;CHECK FOR BLOCK END
 0065 C27800   D     87             JNZ     STOP0              ;IF NOT END THEN JUMP
 0068 23             88             INX     H                  ;  GET BLOCK
 0069 7E             89             MOV     A,M                ;  POINTER MSB
 006A FE03           90             CPI     03H                ;CHECK FOR BLOCK END
 006C C27800   D     91             JNZ     STOP0              ;IF NOT THEN JUMP
 006F CDEC00   D     92             CALL    FRESH0             ;GET FRESH BLOCK
 0072 AF             93             XRA     A                  ;ZERO ACCUMULATER
 0073 02             94             STAX    B                  ;SET BYTE 1
 0074 03             95             INX     B                  ;INCREMENT BLOCK
 0075 02             96             STAX    B                  ;SET BYTE 2
 0076 03             97             INX     B                  ;INCREMENT BLOCK
 0077 02             98             STAX    B                  ;SET BYTE 3
 0078 3A0000   E     99 STOP0:      LDA     STAT               ;GET USER STATUS
 007B E601          100             ANI     01H                ;ISOLATE BIT
 007D FE01          101             CPI     01H                ;USER RDY?
 007F CAAA00   D    102             JZ      STOR0              ;IF SO THEN JUMP
 0082 3A0000   E    103             LDA     SEQ0               ;GET SEQUENCE COUNTER
 0085 FE05          104             CPI     05H                ;STOP STRING RCVD?
 0087 CAB000   D    105             JZ      CRCK0              ;IF YES THEN JUMP
 008A 210500   E    106             LXI     H,TTYBUF+05H       ;GET STOP SEQUENCE ADR
 008D 85            107             ADD     L                  ;ADD OFFSET
 008E 6F            108             MOV     L,A                ;RESET REGISTER
 008F 7E            109 RECK0:      MOV     A,M                ;GET CHARACTER
 0090 BB            110             CMP     E                  ;SAME AS INPUT?
 0091 CAA600   D    111             JZ      INC0               ;IF YES THEN JUMP
 0094 210000   E    112             LXI     H,SEQ0             ;GET SEQUENCE COUNTER
 0097 7E            113             MOV     A,M                ;SET IN ACCUMULATOR
 0098 FE00          114             CPI     00H                ;EQUAL TO 00?
 009A CAAA00   D    115             JZ      STOR0              ;IF YES THEN JUMP
 009D CD5D03   D    116             CALL    RESET              ;RESET SEQUENCE CTR
 00A0 210500   E    117             LXI     H,TTYBUF+05H       ;GET STOP SEQUENCE ADR
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE    3

  LOC  OBJ          LINE        SOURCE STATEMENT

  00A3 C38F00   D    118           JMP     RECK0           ;JUMP
  00A6 210000   E    119 INC0:     LXI     H,SEQ0          ;GET SEQUENCE COUNTER
  00A9 34            120           INR     M               ;INCREMENT COUNTER
  00AA 210000   E    121 STOR0:    LXI     H,BLK0          ;GET BLOCK ADDRESS
  00AD C3D203   D    122           JMP     STORE           ;JUMP
  00B0 7B            123 CRCK0:    MOV     A,E             ;GET CHARACTER
  00B1 FE0D          124           CPI     0DH             ;EQUAL TO <CR>?
  00B3 C2AA00   D    125           JNZ     STOR0           ;IF NOT EQUAL THEN JUMP
  00B6 210000   E    126           LXI     H,STAT          ;GET USER STATUS
  00B9 CDEE03   D    127           CALL    SETSTA          ;SET READY AND STBY
  00BC 010100   E    128           LXI     B,STBY0+01H     ;GET BLOCK ADDRESS
  00BF 0A            129           LDAX    B               ;GET POINTER ADR
  00C0 67            130           MOV     H,A             ;SET IN H REG
  00C1 03            131           INX     B               ;INCREMENT MEMORY
  00C2 0A            132           LDAX    B               ;GET POINTER
  00C3 6F            133           MOV     L,A             ;SET IN L REGISTER
  00C4 36FF          134           MVI     M,0FFH          ;SET REPORT STATUS
  00C6 23            135           INX     H               ;INCREMENT MEM
  00C7 23            136           INX     H               ;INCREMENT MEM
  00C8 23            137           INX     H               ;INCREMENT MEM
  00C9 010000   E    138           LXI     B,STBY0         ;GET STBY COUNTER
  00CC 0A            139           LDAX    B               ;SET IN ACCUM
  00CD 3C            140           INR     A               ;INCREMENT STBY0
  00CE 02            141           STAX    B               ;RESTORE IN STBY0
  00CF 77            142           MOV     M,A             ;SET IN MEMORY
  00D0 210000   E    143           LXI     H,BLK0          ;SET BLOCK COUNTER
  00D3 66            144           MOV     H,M             ;SET IN H REG
  00D4 2E02          145           MVI     L,02H           ;SET BYTE 2 IN L REG
  00D6 010100   E    146           LXI     B,BLK0+01H      ;GET BLOCK POINTER
  00D9 0A            147           LDAX    B               ;SET IN ACCUMULATER
  00DA 77            148           MOV     M,A             ;SET IN BYTE 2
  00DB 03            149           INX     B               ;GET BLOCK POINTER
  00DC 23            150           INX     H               ;INCREMENT MEMORY
  00DD 0A            151           LDAX    B               ;SET IN ACCUMULATER
  00DE 77            152           MOV     M,A             ;SET IN BYTE 2
  00DF 210000   E    153           LXI     H,BLK0          ;GET BLOCK ADDRESS
  00E2 CDD203   D    154           CALL    STORE           ;STORE CHARACTER
  00E5 210000   E    155           LXI     H,SEQ0          ;GET SEQUENCE COUNTER
  00E8 CD0000   E    156           CALL    ZERO1           ;ZERO UTILITY REGISTERS
  00EB C9            157           RET                     ;RETURN
  00EC CD6003   D    158 FRESH0:   CALL    FRESH           ;GET FRESH BLOCK
  00EF 210000   E    159           LXI     H,BLK0          ;GET BLOCK COUNTER
  00F2 7E            160           MOV     A,M             ;SET VALUE IN ACCUM
  00F3 FE00          161           CPI     00H             ;IS THIS FIRST BLOCK?
  00F5 CA0B01   D    162           JZ      ID0             ;IF SO THEN JUMP
  00F8 D5            163           PUSH    D               ;SAVE REGISTER
  00F9 56            164           MOV     D,M             ;SET BLOCK COUNTER IN D
  00FA 1E02          165           MVI     E,02H           ;SET BYTE 2 IN E
  00FC AF            166           XRA     A               ;ZERO ACCUM
  00FD 3D            167           DCR     A               ;DECREMENT ACCUM  (0FFH)
  00FE 12            168           STAX    D               ;STORE FF IN BLOCK POINT
  00FF 13            169           INX     D               ;NEXT BYTE
  0100 3E03          170           MVI     A,03H           ;SET 03 IN ACCUM
  0102 12            171           STAX    D               ;SET IN BLOCK POINT
  0103 86            172           ADD     M               ;  MEMORY
  0104 57            173           MOV     D,A             ;  TO
  0105 AF            174           XRA     A               ;  LAST
  0106 3D            175           DCR     A               ;  BYTE
  0107 5F            176           MOV     E,A             ;  OF
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

0108 78             177           MOV    A,B           ;  THE
0109 12             178           STAX   D             ;  BLOCK
010A D1             179           POP    D             ;RESTORE REGISTER
010B 70             180 ID0:      MOV    M,B           ;  IN BLOCK COUNTER
010C 3E01           181           MVI    A,01H         ;    ID IN
010E 02             182           STAX   B             ;    BYTE 1 OF BLOCK
010F 3E05           183           MVI    A,05H         ;  SET POINTER
0111 010100    E    184           LXI    B,BLK0+01H    ;  TO BYTE
0114 02             185           STAX   B             ;  4 OF
0115 AF             186           XRA    A             ;  OF THE
0116 03             187           INX    B             ;  NEW
0117 02             188           STAX   B             ;  BLOCK
0118 210000    E    189           LXI    H,BLK0        ;GET BLOCK COUNTER
011B 46             190           MOV    B,M           ;SET IN REGISTER
011C 0E01           191           MVI    C,01H         ;SET C REGISTER
011E C9             192           RET                  ;RETURN
                    193 ;
                    194 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    195 ;
                    196 ;     ROUTINE NAME:  USER1
                    197 ;
                    198 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    199 ;
011F DB12           200 USER1:    IN     12H           ;GET CHARACTER FROM PORT 1
0121 E67F           201           ANI    7FH           ;RESET PARITY BIT
0123 5F             202           MOV    E,A           ;STORE INPUT BYTE
0124 3A0001    E    203           LDA    STAT+100H     ;GET USER STATUS
0127 E601           204           ANI    01H           ;ISOLATE BIT 1
0129 FE01           205           CPI    01H           ;IS USER READY?
012B C26401    D    206           JNZ    BLOCK1        ;IF SO THEN CONTINUE
012E 3A0000    E    207 RECHK1:   LDA    SEQ1          ;GET ROUTINE STATUS
0131 210001    E    208           LXI    H,TTYBUF+100H ;GET START SEQUENCE
0134 8D             209           ADC    L             ;SET POINTER
0135 6F             210           MOV    L,A           ;RESTORE POINTER
0136 7E             211           MOV    A,M           ;PUT LETTER IN ACC
0137 210000    E    212           LXI    H,SEQ1        ;LOAD SEQUENCE COUNTER
013A BB             213           CMP    E             ;COMPARE WITH INPUT BYTE
013B CA5401    D    214           JZ     CONT1         ;IF NOT THE SAME THEN JUMP
013E 7E             215           MOV    A,M           ;  COUNTER
013F FE00           216           CPI    00H           ;AT ZERO?
0141 C8             217           RZ                   ;RETURN
0142 CD5D03    D    218           CALL   RESET         ;IF NO THEN RESET COUNTE
0145 210000    E    219           LXI    H,BLK1        ;GET BLOCK COUNTER
0148 46             220           MOV    B,M           ;GET VALUE
0149 AF             221           XRA    A             ;ZERO OUT ACCUMULATER
014A 4F             222           MOV    C,A           ;ZERO OUT C REGISTER
014B 02             223           STAX   B             ;ZERO OUT BYTE 1 IN BLOCK
014C 77             224           MOV    M,A           ;ZERO OUT BLOCK COUNTER
014D 23             225           INX    H             ;GET BLOCK POINTER LSB
014E 77             226           MOV    M,A           ;ZERO OUT BLOCK POINTER
014F 23             227           INX    H             ;GET BLOCK POINTER MSB
0150 77             228           MOV    M,A           ;ZERO OUT BLOCK POINTER
0151 C32E01    D    229           JMP    RECHK1        ;JUMP TO RECHECK CHARACTER
0154 7E             230 CONT1:    MOV    A,M           ;  COUNTER
0155 3C             231           INR    A             ;  COUNTER
0156 77             232           MOV    M,A           ;SET IN MEMORY
0157 FE05           233           CPI    05H           ;CHECK FOR FULL COUNT
0159 C26401    D    234           JNZ    BLOCK1        ;IF NOT FULL THEN JUMP
015C 210001    E    235           LXI    H,STAT+100H   ;  TO RCVG
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE    5

  LOC  OBJ         LINE        SOURCE STATEMENT

  015F 34           236           INR   M               ;SET TO RCVG
  0160 AF           237           XRA   A               ;ZERO ACCUMULATER
  0161 320000   E   238           STA   SEQ1            ;SET SEQUENCE COUNTER TO ZERO
  0164 3A0000   E   239 BLOCK1:   LDA   BLK1            ;GET BLOCK COUNTER
  0167 FE00         240           CPI   00H             ;BLOCK ASSIGNED?
  0169 C27E01   D   241           JNZ   POINT1          ;IF YES THEN JUMP
  016C CD0B02   D   242           CALL  FRESH1          ;GET FRESH BLOCK
  016F 210100   E   243           LXI   H,STBY1+01H     ;GET REGISTER
  0172 70           244           MOV   M,B             ;  SET ADDRESS
  0173 23           245           INX   H               ;  IN B
  0174 71           246           MOV   M,C             ;  REGISTER PAIR
  0175 AF           247           XRA   A               ;ZERO ACCUMULATER
  0176 02           248           STAX  B               ;SET BYTE 1
  0177 03           249           INX   B               ;INCREMENT BLOCK
  0178 02           250           STAX  B               ;SET BYTE 2
  0179 03           251           INX   B               ;INCREMENT BLOCK
  017A 02           252           STAX  B               ;SET BYTE 3
  017B C39701   D   253           JMP   STOP1           ;JUMP
  017E 210200   E   254 POINT1:   LXI   H,SEQ1+2                ;  GET BLOCK
  0181 7E           255           MOV   A,M             ;  POINTER LSB
  0182 FEFF         256           CPI   0FFH            ;CHECK FOR BLOCK END
  0184 C29701   D   257           JNZ   STOP1           ;IF NOT END THEN JUMP
  0187 23           258           INX   H               ;  GET BLOCK
  0188 7E           259           MOV   A,M             ;  POINTER MSB
  0189 FE03         260           CPI   03H             ;CHECK FOR BLOCK END
  018B C29701   D   261           JNZ   STOP1           ;IF NOT THEN JUMP
  018E CD0B02   D   262           CALL  FRESH1          ;GET FRESH BLOCK
  0191 AF           263           XRA   A               ;ZERO ACCUMULATER
  0192 02           264           STAX  B               ;SET BYTE 1
  0193 03           265           INX   B               ;INCREMENT BLOCK
  0194 02           266           STAX  B               ;SET BYTE 2
  0195 03           267           INX   B               ;INCREMENT BLOCK
  0196 02           268           STAX  B               ;SET BYTE 3
  0197 3A0001   E   269 STOP1:    LDA   STAT+100H               ;GET USER STATUS
  019A E601         270           ANI   01H             ;ISOLATE BIT
  019C FE01         271           CPI   01H             ;USER RDY?
  019E CAC901   D   272           JZ    STOR1           ;IF SO THEN JUMP
  01A1 3A0000   E   273           LDA   SEQ1            ;GET SEQUENCE COUNTER
  01A4 FE05         274           CPI   05H             ;STOP STRING RCVD?
  01A6 CACF01   D   275           JZ    CRCK1           ;IF YES THEN JUMP
  01A9 210501   E   276           LXI   H,TTYBUF+105H   ;GET STOP SEQUENCE ADR
  01AC 85           277           ADD   L               ;ADD OFFSET
  01AD 6F           278           MOV   L,A             ;RESET REGISTER
  01AE 7E           279 RECK1:    MOV   A,M             ;GET CHARACTER
  01AF BB           280           CMP   E               ;SAME AS INPUT?
  01B0 CAC501   D   281           JZ    INC1            ;IF YES THEN JUMP
  01B3 210000   E   282           LXI   H,SEQ1          ;GET SEQUENCE COUNTER
  01B6 7E           283           MOV   A,M             ;SET IN ACCUMULATOR
  01B7 FE00         284           CPI   00H             ;EQUAL TO 00?
  01B9 CAC901   D   285           JZ    STOR1           ;IF YES THEN JUMP
  01BC CD5D03   D   286           CALL  RESET           ;RESET SEQUENCE CTR
  01BF 210501   E   287           LXI   H,TTYBUF+105H   ;GET STOP SEQUENCE ADDRESS
  01C2 C3AE01   D   288           JMP   RECK1           ;JUMP
  01C5 210000   E   289 INC1:     LXI   H,SEQ1          ;GET SEQUENCE COUNTER
  01C8 34           290           INR   M               ;INCREMENT COUNTER
  01C9 210000   E   291 STOR1:    LXI   H,BLK1          ;GET BLOCK ADDRESS
  01CC C3D203   D   292           JMP   STORE           ;JUMP
  01CF 7B           293 CRCK1:    MOV   A,E             ;GET CHARACTER
  01D0 FE0D         294           CPI   0DH             ;EQUAL TO <CR>?
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE    6

  LOC  OBJ           LINE        SOURCE STATEMENT

  01D2 C2C901    D    295         JNZ     STOR1           ;IF NOT EQUAL THEN JUMP
  01D5 210001    E    296         LXI     H,STAT+100H     ;GET USER STATUS
  01D8 CDEE03    D    297         CALL     SETSTA           ;SET READY AND STBY
  01DB 010100    E    298         LXI     B,STBY1+01H     ;GET BLOCK ADDRESS
  01DE 0A             299         LDAX    B               ;GET POINTER ADR
  01DF 67             300         MOV     H,A             ;SET IN H REGISTER
  01E0 03             301         INX     B               ;INCREMENT MEMORY
  01E1 0A             302         LDAX    B               ;GET POINTER ADR
  01E2 6F             303         MOV     L,A             ;SET IN L REGISTER
  01E3 36FF           304         MVI     M,0FFH          ;SET REPORT STATUS
  01E5 23             305         INX     H               ;INCREMENT MEM
  01E6 23             306         INX     H               ;INCREMENT MEM
  01E7 23             307         INX     H               ;INCREMENT MEM
  01E8 010000    E    308         LXI     B,STBY1         ;GET STBY COUNTER
  01EB 0A             309         LDAX    B               ;SET IN ACCUM
  01EC 3C             310         INR     A               ;INCREMENT STBY1
  01ED 02             311         STAX    B               ;RESTORE IN STBY1
  01EE 77             312         MOV     M,A             ;SET IN MEMORY
  01EF 210000    E    313         LXI     H,BLK1          ;SET BLOCK COUNTER
  01F2 66             314         MOV     H,M             ;SET IN H REG
  01F3 2E02           315         MVI     L,02H           ;SET BYTE 2 IN L REG
  01F5 010100    E    316         LXI     B,BLK1+01H      ;GET BLOCK POINTER
  01F8 0A             317         LDAX    B               ;SET IN ACCUMULATER
  01F9 77             318         MOV     M,A             ;SET IN BYTE 2
  01FA 03             319         INX     B               ;INCREMENT BLOCK POINTER
  01FB 23             320         INX     H               ;INCREMENT MEMORY
  01FC 0A             321         LDAX    B               ;SET IN ACCUMULATER
  01FD 77             322         MOV     M,A             ;SET IN BYTE 2
  01FE 210000    E    323         LXI     H,BLK1          ;GET BLOCK ADDRESS
  0201 CDD203    D    324         CALL    STORE           ;STORE CHARACTER
  0204 210000    E    325         LXI     H,SEQ1          ;GET SEQUENCE COUNTER
  0207 CD0000    E    326         CALL    ZERO1           ;ZERO UTILITY REGISTERS
  020A C9             327         RET                     ;RETURN
  020B CD6003    D    328 FRESH1: CALL    FRESH           ;GET FRESH BLOCK
  020E 210000    E    329         LXI     H,BLK1          ;GET BLOCK COUNTER
  0211 7E             330         MOV     A,M             ;SET VALUE IN ACCUM
  0212 FE00           331         CPI     00H             ;IS THIS FIRST BLOCK?
  0214 CA2A02    D    332         JZ      ID1             ;IF SO THEN JUMP
  0217 D5             333         PUSH    D               ;SAVE REGISTER
  0218 56             334         MOV     D,M             ;SET BLOCK COUNTER IN D
  0219 1E02           335         MVI     E,02H           ;SET BYTE 2 IN E
  021B AF             336         XRA     A               ;ZERO ACCUM
  021C 3D             337         DCR     A               ;DECREMENT ACCUM(0FFH)
  021D 12             338         STAX    D               ;STORE FF IN BLOCK POINT
  021E 13             339         INX     D               ;NEXT BYTE
  021F 3E03           340         MVI     A,03H           ;SET 03 IN ACCUM
  0221 12             341         STAX    D               ;SET IN BLOCK POINT
  0222 86             342         ADD     M               ;  MEMORY
  0223 57             343         MOV     D,A             ;  TO
  0224 AF             344         XRA     A               ;  LAST
  0225 3D             345         DCR     A               ;  BYTE
  0226 5F             346         MOV     E,A             ;  OF
  0227 78             347         MOV     A,B             ;  THE
  0228 12             348         STAX    D               ;  BLOCK
  0229 D1             349         POP     D               ;RESTORE REGISTER
  022A 70             350 ID1:    MOV     M,B             ;  IN BLOCK COUNTER
  022B 3E02           351         MVI     A,02H           ;     SET USER ID IN
  022D 02             352         STAX    B               ;     BYTE 1 OF BLOCK
  022E 3E05           353         MVI     A,05H           ;SET POINTER
```

```
 LOC  OBJ         LINE        SOURCE STATEMENT

 0230 010100   E   354          LXI     B,BLK1+01H      ;  SET POINTER TO BYTE
 0233 02           355          STAX    B               ;  4 OF
 0234 AF           356          XRA     A               ;  OF THE
 0235 03           357          INX     B               ;  NEW
 0236 02           358          STAX    B               ;  BLOCK
 0237 210000   E   359          LXI     H,BLK1          ;GET BLOCK COUNTER
 023A 46           360          MOV     B,M             ;SET IN REGISTER
 023B 0E01         361          MVI     C,01H           ;SET C REGISTER
 023D C9           362          RET                     ;RETURN
                   363 ;
                   364 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   365 ;
                   366 ;     ROUTINE NAME:  USER2
                   367 ;
                   368 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   369 ;
 023E DB14         370 USER2:   IN      14H             ;GET CHARACTER FROM PORT 2
 0240 E67F         371          ANI     7FH             ;RESET PARITY BIT
 0242 5F           372          MOV     E,A             ;STORE INPUT BYTE
 0243 3A0002   E   373          LDA     STAT+200H       ;GET USER STATUS
 0246 E601         374          ANI     01H             ;ISOLATE BIT 1
 0248 FE01         375          CPI     01H             ;IS USER READY?
 024A C28302   D   376          JNZ     BLOCK2          ;IF SO THEN JUMP
 024D 3A0000   E   377 RECHK2:  LDA     SEQ2            ;GET ROUTINE STATUS
 0250 210002   E   378          LXI     H,TTYBUF+200H   ;GET START SEQUENCE
 0253 8D           379          ADC     L               ;SET POINTER
 0254 6F           380          MOV     L,A             ;RESTORE POINTER
 0255 7E           381          MOV     A,M             ;PUT LETTER IN ACC
 0256 210000   E   382          LXI     H,SEQ2          ;GET SEQUENCE COUNTER
 0259 BB           383          CMP     E               ;COMPARE WITH INPUT
 025A CA7302   D   384          JZ      CONT2           ;IF NOT THE SAME THEN JUMP
 025D 7E           385          MOV     A,M             ;  COUNTER
 025E FE00         386          CPI     00H             ;AT ZERO?
 0260 C8           387          RZ                      ;RETURN
 0261 CD5D03   D   388          CALL    RESET           ;IF NO THEN RESET COUNTER
 0264 210000   E   389          LXI     H,BLK2          ;GET BLOCK COUNTER
 0267 46           390          MOV     B,M             ;GET VALUE
 0268 AF           391          XRA     A               ;ZERO OUT ACCUMULATER
 0269 4F           392          MOV     C,A             ;ZERO OUT C REGISTER
 026A 02           393          STAX    B               ;ZERO OUT BYTE 1 IN BLOCK
 026B 77           394          MOV     M,A             ;ZERO OUT BLOCK COUNTER
 026C 23           395          INX     H               ;GET BLOCK POINTER LSB
 026D 77           396          MOV     M,A             ;ZERO OUT BLOCK POINTER
 026E 23           397          INX     H               ;GET BLOCK POINTER MSB
 026F 77           398          MOV     M,A             ;ZERO OUT BLOCK POINTER
 0270 C34D02   D   399          JMP     RECHK2          ;JUMP TO RECHECK CHARACTER
 0273 7E           400 CONT2:   MOV     A,M             ;  SEQUENCE
 0274 3C           401          INR     A               ;  COUNTER
 0275 77           402          MOV     M,A             ;SET IN MEMORY
 0276 FE05         403          CPI     05H             ;CHECK FOR FULL COUNT
 0278 C28302   D   404          JNZ     BLOCK2          ;IF NOT FULL THEN JUMP
 027B 210002   E   405          LXI     H,STAT+200H     ;  ZERO
 027E 34           406          INR     M               ;SET TO RCVG
 027F AF           407          XRA     A               ;  COUNTER
 0280 320000   E   408          STA     SEQ2            ;SET SEQUENCE COUNTER TO ZERO
 0283 3A0000   E   409 BLOCK2:  LDA     BLK2            ;GET BLOCK COUNTER
 0286 FE00         410          CPI     00H             ;BLOCK ASSIGNED?
 0288 C29D02   D   411          JNZ     POINT2          ;IF YES THEN JUMP
 028B CD2A03   D   412          CALL    FRESH2          ;GET FRESH BLOCK
 LOC  OBJ         LINE        SOURCE STATEMENT

028E 210100   E   413         LXI    H,STBY2+01H      ;GET REGISTER
0291 70           414         MOV    M,B              ;  SET ADDRESS
0292 23           415         INX    H                ;  IN B
0293 71           416         MOV    M,C              ;  REGISTER PAIR
0294 AF           417         XRA    A                ;ZERO ACCUMULATER
0295 02           418         STAX   B                ;SET BYTE 1
0296 03           419         INX    B                ;INCREMENT BLOCK
0297 02           420         STAX   B                ;SET BYTE 2
0298 03           421         INX    B                ;INCREMENT BLOCK
0299 02           422         STAX   B                ;SET BYTE 3
029A C3B602   D   423         JMP    STOP2            ;JUMP
029D 210200   E   424 POINT2: LXI    H,SEQ2+2                ;  GET BLOCK
02A0 7E           425         MOV    A,M              ;  POINTER LSB
02A1 FEFF         426         CPI    0FFH             ;CHECK FOR BLOCK END
02A3 C2B602   D   427         JNZ    STOP2            ;IF NOT END THEN JUMP
02A6 23           428         INX    H                ;  GET BLOCK
02A7 7E           429         MOV    A,M              ;  POINTER MSB
02A8 FE03         430         CPI    03H              ;CHECK FOR BLOCK END
02AA C2B602   D   431         JNZ    STOP2            ;IF NOT THEN JUMP
02AD CD2A03   D   432         CALL   FRESH2           ;GET FRESH BLOCK
02B0 AF           433         XRA    A                ;ZERO ACCUMULATER
02B1 02           434         STAX   B                ;SET BYTE 1
02B2 03           435         INX    B                ;INCREMENT BLOCK
02B3 02           436         STAX   B                ;SET BYTE 2
02B4 03           437         INX    B                ;INCREMENT BLOCK
02B5 02           438         STAX   B                ;SET BYTE 3
02B6 3A0002   E   439 STOP2:  LDA    STAT+200H        ;GET USER STATUS
02B9 E601         440         ANI    01H              ;ISOLATE BIT
02BB FE01         441         CPI    01H              ;USER RDY?
02BD CAE802   D   442         JZ     STOR2            ;IF SO THEN JUMP
02C0 3A0000   E   443         LDA    SEQ2             ;GET SEQUENCE COUNTER
02C3 FE05         444         CPI    05H              ;STOP STRING RCVD?
02C5 CAEE02   D   445         JZ     CRCK2            ;IF YES THEN JUMP
02C8 210502   E   446         LXI    H,TTYBUF+205H    ;GET STOP SEQUENCE ADR
02CB 85           447         ADD    L                ;ADD OFFSET
02CC 6F           448         MOV    L,A              ;RESET REGISTER
02CD 7E           449 RECK2:  MOV    A,M              ;GET CHARACTER
02CE BB           450         CMP    E                ;SAME AS INPUT?
02CF CAE402   D   451         JZ     INC2             ;IF YES THEN JUMP
02D2 210000   E   452         LXI    H,SEQ2           ;GET SEQUENCE COUNTER
02D5 7E           453         MOV    A,M              ;SET IN ACCUMULATOR
02D6 FE00         454         CPI    00H              ;EQUAL TO 00?
02D8 CAE802   D   455         JZ     STOR2            ;IF YES THEN JUMP
02DB CD5D03   D   456         CALL   RESET            ;RESET SEQUENCE CTR
02DE 210502   E   457         LXI    H,TTYBUF+205H    ;GET STOP SEQUENCE ADDRESS
02E1 C3CD02   D   458         JMP    RECK2            ;JUMP
02E4 210000   E   459 INC2:   LXI    H,SEQ2           ;GET SEQUENCE COUNTER
02E7 34           460         INR    M                ;INCREMENT COUNTER
02E8 210000   E   461 STOR2:  LXI    H,BLK2           ;GET BLOCK COUNTER
02EB C3D203   D   462         JMP    STORE            ;JUMP
02EE 7B           463 CRCK2:  MOV    A,E              ;GET CHARACTER
02EF FE0D         464         CPI    0DH              ;EQUAL TO <CR>?
02F1 C2E802   D   465         JNZ    STOR2            ;IF NOT EQUAL THEN JUMP
02F4 210002   E   466         LXI    H,STAT+200H      ;GET USER STATUS
02F7 CDEE03   D   467         CALL    SETSTA
                                                         ;SET READY AND STBY
02FA 010100   E   468         LXI    B,STBY2+01H      ;GET BLOCK ADDRESS
02FD 0A           469         LDAX   B                ;GET POINTER ADR
02FE 67           470         MOV    H,A              ;SET IN H REGISTER
02FF 03           471         INX    B                ;INCREMENT MEMORY
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE    9

  LOC OBJ         LINE        SOURCE STATEMENT

  0300 0A          472          LDAX    B              ;GET POINTER ADR
  0301 6F          473          MOV     L,A            ;SET IN L REGISTER
  0302 36FF        474          MVI     M,0FFH         ;SET REPORT STATUS
  0304 23          475          INX     H              ;INCREMENT MEM
  0305 23          476          INX     H              ;INCREMENT MEM
  0306 23          477          INX     H              ;INCREMENT MEM
  0307 010000   E  478          LXI     B,STBY2        ;GET STBY COUNTER
  030A 0A          479          LDAX    B              ;SET IN ACCUM
  030B 3C          480          INR     A              ;INCREMENT STBY2
  030C 02          481          STAX    B              ;RESTORE IN STBY2
  030D 77          482          MOV     M,A            ;SET IN MEMORY
  030E 210000   E  483          LXI     H,BLK2         ;SET BLOCK COUNTER
  0311 66          484          MOV     H,M            ;SET IN H REG
  0312 2E02        485          MVI     L,02H          ;SET BYTE 2 IN L REG
  0314 010100   E  486          LXI     B,BLK2+01H     ;GET BLOCK POINTER
  0317 0A          487          LDAX    B              ;SET IN ACCUMULATER
  0318 77          488          MOV     M,A            ;SET IN BYTE 2
  0319 03          489          INX     B              ;GET BLOCK POINTER
  031A 23          490          INX     H              ;INCREMENT MEMORY
  031B 0A          491          LDAX    B              ;SET IN ACCUMULATER
  031C 77          492          MOV     M,A            ;SET IN BYTE 2
  031D 210000   E  493          LXI     H,BLK2         ;GET BLOCK ADDRESS
  0320 CDD203   D  494          CALL    STORE          ;STORE CHARACTER
  0323 210000   E  495          LXI     H,SEQ2         ;GET SEQUENCE COUNTER
  0326 CD0000   E  496          CALL    ZERO1          ;ZERO UTILITY REGISTERS
  0329 C9          497          RET                    ;RETURN
  032A CD6003   D  498 FRESH2:  CALL    FRESH          ;GET FRESH BLOCK
  032D 210000   E  499          LXI     H,BLK2         ;GET BLOCK COUNTER
  0330 7E          500          MOV     A,M            ;SET VALUE IN ACCUMULATER
  0331 FE00        501          CPI     00H            ;IS THIS FIRST BLOCK?
  0333 CA4903   D  502          JZ      ID2            ;IF SO THEN JUMP
  0336 D5          503          PUSH    D              ;SAVE REGISTER
  0337 56          504          MOV     D,M            ;SET BLOCK COUNTER IN D
  0338 1E02        505          MVI     E,02H          ;SET BYTE 2 IN E
  033A AF          506          XRA     A              ;ZERO ACCUM
  033B 3D          507          DCR     A              ;DECREMENTACCUM (0FFH)
  033C 12          508          STAX    D              ;STORE FF IN BLOCK POINT
  033D 13          509          INX     D              ;NEXT BYTE
  033E 3E03        510          MVI     A,03H          ;SET 03 IN ACCUM
  0340 12          511          STAX    D              ;SET IN BLOCK POINT
  0341 86          512          ADD     M              ;  MEMORY
  0342 57          513          MOV     D,A            ;  TO
  0343 AF          514          XRA     A              ;  LAST
  0344 3D          515          DCR     A              ;  BYTE
  0345 5F          516          MOV     E,A            ;  OF
  0346 78          517          MOV     A,B            ;  THE
  0347 12          518          STAX    D              ;  BLOCK
  0348 D1          519          POP     D              ;RESTORE REGISTER
  0349 70          520 ID2:     MOV     M,B            ;  IN BLOCK COUNTER
  034A 3E03        521          MVI     A,03H          ;    SET USER ID IN
  034C 02          522          STAX    B              ;    BYTE 1 OF BLOCK
  034D 3E05        523          MVI     A,05H          ;  SET POINTER
  034F 010100   E  524          LXI     B,BLK2+01H     ;  TO BYTE
  0352 02          525          STAX    B              ;  4 OF
  0353 AF          526          XRA     A              ;  OF THE
  0354 03          527          INX     B              ;  NEW
  0355 02          528          STAX    B              ;  BLOCK
  0356 210000   E  529          LXI     H,BLK2         ;GET BLOCK COUNTER
  0359 46          530          MOV     B,M            ;GET COUNTER
```

TYPING GUIDE SHEET

```
  LOC  OBJ         LINE        SOURCE STATEMENT

  035A 0E01         531          MVI      C,01H          ;SET C REGISTER
  035C C9           532          RET                     ;RETURN
                    533 ;
                    534 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    535 ;
                    536 ;      ROUTINE NAME: RESET
                    537 ;      INPUTS: SEQUENCE COUNTER ADDRESS IN HL REGISTER PAIR
                    538 ;      OUTPUTS: NONE
                    539 ;      CALLS: NONE
                    540 ;      DESCRIPTION: CLEARS A REGISTER
                    541 ;
                    542 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    543 ;
  035D AF           544 RESET:   XRA      A              ;  RESET
  035E 77           545          MOV      M,A            ;SET TO ZERO
  035F C9           546          RET                     ;RETURN
                    547 ;
                    548 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    549 ;
                    550 ;      ROUTINE NAME:  FRESH
                    551 ;      INPUTS: NONE
                    552 ;      OUTPUTS: ADDRESS IN THE BC REG PAIR
                    553 ;      CALLS: OPER
                    554 ;      DESCRIPTION: THIS SUBROUTINE SEARCHES FOR A CLEAR BLOCK OF STORAGE
                    555 ;                   AND RETURNS IT.  IF THERE ARE NO CLEAR BLOCKS, THE
                    556 ;                   PROGRAM INITIATES ERROR PROCEDINGS.
                    557 ;
                    558 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    559 ;
  0360 210000   E   560 FRESH:   LXI      H,BOTTOM       ;LOAD ADDRESS OF BOTTOM INTO H-L-REG
  0363 3A0000   E   561 NXTBLK:  LDA      TOP            ;LOAD TOP-MSB INTO A-REG
  0366 BC           562          CMP      H              ;IS H-L-REG POINTING TO TOP?
  0367 CA9D03   D   563          JZ       FULL           ;IF YES, JUMP TO FULL
  036A 7E           564          MOV      A,M            ;GET BLOCK STATUS
  036B FE00         565          CPI      00H            ;IS BLOCK EMPTY?
  036D CA7703   D   566          JZ       RESUME         ;IF YES, JUMP TO RESUME
  0370 3E04         567          MVI      A,04H          ;PUT 4 INTO A-REG
  0372 8C           568          ADC      H              ;ADD H-REG TO A-REG
  0373 67           569          MOV      H,A            ;MOVE SUM TO H-REG TO INCREMENT TO NEXT BLOCK
  0374 C36303   D   570          JMP      NXTBLK         ;JUMP BACK TO NXTBLK
  0377 3A0000   E   571 RESUME:  LDA      OVRFLW         ;LOAD OVRFLW INTO A-REG
  037A FE00         572          CPI      00H            ;IS OVRFLW EQUAL TO ZERO?
  037C CA9A03   D   573          JZ       FOUND          ;IF YES, JUMP TO FOUND
  037F E5           574          PUSH     H              ;STORE CONTENTS OF H-L-REG
  0380 F3           575          DI                      ;DISABLE ALL INTERRUPTS
  0381 DB19         576          IN       19H            ;READ CURRENT INTERRUPT MASK BYTE FROM PIC-0
  0383 E6EA         577          ANI      0EAH           ;RESET BITS FOR INSTRUMENT INTERRUPT MASKS
  0385 D319         578          OUT      19H            ;PROGRAM PIC-0 TO UNMASK INSTRUMENT INTERRUPTS
  0387 3A0000   E   579          LDA      SYSTAT         ;LOAD SYSTAT INTO A-REG
  038A E6FE         580          ANI      0FEH           ;RESET MEMORY OVERFLOW ERROR BITY
  038C 320000   E   581          STA      SYSTAT         ;UPDATE SYSTAT IN MEMORY
  038F 210000   E   582          LXI      H,VECFLG       ;LOAD ADDRESS OF VECFLG INTO H-L-REG
  0392 36FF         583          MVI      M,0FFH         ;SET VECFLG TO FF (DARK VECTOR MODE)
  0394 210000   E   584          LXI      H,OVRFLW       ;LOAD ADDRESS OF OVRFLW INTO H-L-REG
  0397 3600         585          MVI      M,00H          ;RESET VECFLG TO ZERO
  0399 E1           586          POP      H              ;RESTORE CONTENTS OF H-L-REG
  039A 44           587 FOUND:   MOV      B,H            ;     COPY ADDRESS OF EMPTY BLOCK FROM
  039B 4D           588          MOV      C,L            ;     H-L-REG TO B-C-REG
  039C C9           589          RET                     ;RETURN TO INVOKING PROCEDURE
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0          USER     PAGE   11

LOC  OBJ       LINE        SOURCE STATEMENT

039D 3EC0       590 FULL:   MVI     A,0C0H          ;SET ERROR CODE FOR ERROR AND STOP DATA
039F D310       591         OUT     10H             ;STOP USER 0
03A1 D312       592         OUT     12H             ;STOP USER 1
03A3 D314       593         OUT     14H             ;STOP USER 2
03A5 3A0000  E  594         LDA     OVRFLW          ;LOAD OVRFLW INTO A-REG
03A8 FEFF       595         CPI     0FFH            ;IS OVRFLW SET TO FF?
03AA CACC03  D  596         JZ      UPDATE          ;IF YES, JUMP TO UPDATE
03AD DB19       597         IN      19H             ;READ CURRENT INTERRUPT MASK BYTE FROM PIC-0
03AF F615       598         ORI     015H            ;SET BITS FOR INSTRUMENT INTERRUPT MASKS
03B1 D319       599         OUT     19H             ;PROGRAM PIC-0 TO MASK INSTRUMENT INTERRUPTS
03B3 CD0000  E  600         CALL    EOI534          ;RESET PICs ON 534 BOARD
03B6 CD0000  E  601         CALL    EOIM            ;RESET PICs ON MAIN BOARD
03B9 3A0000  E  602         LDA     SYSTAT          ;LOAD SYSTAT INTO A-REG
03BC F601       603         ORI     01H             ;SET MEMORY OVERFLOW ERROR BIT
03BE 320000  E  604         STA     SYSTAT          ;UPDATE SYSTAT IN MEMORY
03C1 0E07       605         MVI     C,07H           ;SET BELL CHARACTER IN D-REG
03C3 CD0000  E  606         CALL    CO              ;SEND BELL TO CONSOLE TO WIPE OUT ANY HALF-DONE ESC
                    APE SEQUENCE
03C6 3EFF       607         MVI     A,0FFH          ;MOVE FF INTO A-REG
03C8 320000  E  608         STA     OVRFLW          ;SET OVRFLW TO FF
03CB FB         609         EI                      ;ENABLE INTERRUPTS TO ALLOW FOR NOVA COMMUNICATION
03CC CD0000  E  610 UPDATE: CALL    OPER            ;UPDATE CONSOLE (THIS INCLUDES CALLS TO 'NOVA')
03CF C36003  D  611         JMP     FRESH           ;JUMP BACK TO FRESH
                612 ;
                613 ;-:-:-:-:-:-:-:~:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                614 ;
                615 ;       ROUTINE NAME: STORE
                616 ;       INPUTS: H,L,D
                617 ;       OUTPUTS: NONE
                618 ;       CALLS: NONE
                619 ;       DESCRIPTION: PROGRAM STORES LETTER GIVEN MEMORY LOCATION
                620 ;
                621 ;-:-:-:-:-:-:-:~:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                622 ;
03D2 46         623 STORE:  MOV     B,M             ;SET IN B REGISTER
03D3 AF         624         XRA     A               ;ZERO ACCUMULATER
03D4 4F         625         MOV     C,A             ;SET IN REGISTER C
03D5 23         626         INX     H               ;GET BLOCK POINTER LSB
03D6 E5         627         PUSH    H               ;SAVE ADDRESS
03D7 C5         628         PUSH    B               ;STORE BLOCK ADDRESS
03D8 4E         629         MOV     C,M             ;SET IN C REGISTER
03D9 23         630         INX     H               ;GET BLOCK POINTER MSB
03DA 46         631         MOV     B,M             ;SET IN B REGISTER
03DB C5         632         PUSH    B               ;STORE REGISTER
03DC E1         633         POP     H               ;GET BLOCK COUNTER
03DD C1         634         POP     B               ;GET  BLOCK POINTER
03DE 7D         635         MOV     A,L             ;  UPDATE
03DF 89         636         ADC     C               ;  POINTER
03E0 4F         637         MOV     C,A             ;  LSB
03E1 7C         638         MOV     A,H             ;    UPDATE
03E2 88         639         ADC     B               ;    POINTER
03E3 47         640         MOV     B,A             ;    MSB
03E4 7B         641         MOV     A,E             ;PUT CHARACTER IN ACC
03E5 02         642         STAX    B               ;STORE IN MEMORY
03E6 23         643         INX     H               ;INCREMENT POINTER
03E7 C1         644         POP     B               ;GET REG FOR BLOCK POINT LSB
03E8 7D         645         MOV     A,L             ;GET BLOCK POINT LSB
03E9 02         646         STAX    B               ;STORE BLOCK POINTER LSB
03EA 03         647         INX     B               ;GET REG FOR BLOCK POINT MSB
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        USER     PAGE   12

  LOC  OBJ         LINE        SOURCE STATEMENT

 03EB 7C            648          MOV     A,H              ;GET BLOCK POINT MSB
 03EC 02            649          STAX    B                ;STORE BLOCK POINTER MSB
 03ED C9            650          RET                      ;RETURN
                    651 ;
                    652 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-;
                    653 ;
                    654 ;        ROUTINE NAME:  SETSTA
                    655 ;        INPUTS:  ADDRESS OF STATUS BYTE IN H-L-REG
                    656 ;        OUTPUTS:  STATUS BYTE IN MEMORY
                    657 ;        DESCRIPTION:  SETS BITS FOR READY AND STANDBY IN STATUS BYTE
                    658 ;
                    659 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-;
                    660 ;
 03EE F5            661 SETSTA: PUSH   PSW              ;STORE A-REG AND FLAGS IN STACK
 03EF 7E            662         MOV    A,M              ;MOVE STATUS BYTE TO A-REG
 03F0 F605          663         ORI    05H              ;SET BITS FOR READY AND STANDBY
 03F2 E6FD          664         ANI    0FDH             ;RESET BIT FOR RECEIVING
 03F4 77            665         MOV    M,A              ;UPDATE STATUS BYTE IN MEMORY
 03F5 F1            666         POP    PSW              ;RESTORE A-REG AND FLAGS FROM STACK
 03F6 C9            667         RET
                    668         END                     ;END OF MODULE

PUBLIC SYMBOLS
USER0  D 0000    USER1  D 011F    USER2  D 023E

EXTERNAL SYMBOLS
BLK0   E 0000    BLK1   E 0000    BLK2   E 0000    BOTTOM E 0000    C0     E 0000    EOI534 E 0000
EOIM   E 0000    OPER   E 0000    OVRFLW E 0000    SEQ0   E 0000    SEQ1   E 0000    SEQ2   E 0000
STAT   E 0000    STBY0  E 0000    STBY1  E 0000    STBY2  E 0000    SYSTAT E 0000    TOP    E 0000
TTYBUF E 0000    VECFLG E 0000    ZERO1  E 0000

USER SYMBOLS
BLK0   E 0000    BLK1   E 0000    BLK2   E 0000    BLOCK0 D 0045    BLOCK1 D 0164    BLOCK2 D 0283
BOTTOM E 0000    C0     E 0000    CONT0  D 0035    CONT1  D 0154    CONT2  D 0273    CRCK0  D 00B0
CRCK1  D 01CF    CRCK2  D 02EE    EOI534 E 0000    EOIM   E 0000    FOUND  D 039A    FRESH  D 0360
FRESH0 D 00EC    FRESH1 D 020B    FRESH2 D 032A    FULL   D 039D    ID0    D 010B    ID1    D 022A
ID2    D 0349    INC0   D 00A6    INC1   D 01C5    INC2   D 02E4    NXTBLK D 0363    OPER   E 0000
OVRFLW E 0000    POINT0 D 005F    POINT1 D 017E    POINT2 D 029D    RECHK0 D 000F    RECHK1 D 012E
RECHK2 D 024D    RECK0  D 008F    RECK1  D 01AE    RECK2  D 02CD    RESET  D 035D    RESUME D 0377
SEQ0   E 0000    SEQ1   E 0000    SEQ2   E 0000    SETSTA D 03EE    STAT   E 0000    STBY0  E 0000
STBY1  E 0000    STBY2  E 0000    STOP0  D 0078    STOP1  D 0197    STOP2  D 02B6    STOR0  D 00AA
STOR1  D 01C9    STOR2  D 02E8    STORE  D 03D2    SYSTAT E 0000    TOP    E 0000    TTYBUF E 0000
UPDATE D 03CC    USER0  D 0000    USER1  D 011F    USER2  D 023E    VECFLG E 0000    ZERO1  E 0000

ASSEMBLY COMPLETE,   NO ERRORS
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                  PAGE   1

BLK0      17     49     69    121    143    146    153    159    184    189
BLK1      17    219    239    291    313    316    323    329    354    359
BLK2      17    389    409    461    483    486    493    499    524    529
BLOCK0    36     64     69#
BLOCK1   206    234    239#
BLOCK2   376    404    409#
BOTTOM    17    560
CO        16    606
CONT0     44     60#
CONT1    214    230#
CONT2    384    400#
CRCK0    105    123#
CRCK1    275    293#
CRCK2    445    463#
EOI534    16    600
EOIM      16    601
FOUND    573    587#
FRESH    158    328    498    560#   611
FRESH0    72     92    158#
FRESH1   242    262    328#
FRESH2   412    432    498#
FULL     563    590#
ID0      162    180#
ID1      332    350#
ID2      502    520#
INC0     111    119#
INC1     281    289#
INC2     451    459#
NXTBLK   561#   570
OPER      17    610
OVRFLW    18    571    584    594    608
POINT0    71     84#
POINT1   241    254#
POINT2   411    424#
RECHK0    37#    59
RECHK1   207#   229
RECHK2   377#   399
RECK0    109#   118
RECK1    279#   288
RECK2    449#   458
RESET     48    116    218    286    388    456    544#
RESUME   566    571#
SEQ0      17     37     42     68     84    103    112    119    155
SEQ1      17    207    212    238    254    273    282    289    325
SEQ2      17    377    382    408    424    443    452    459    495
SETSTA   127    297    467    661#
STAT      17     33     65     99    126    203    235    269    296    373    405    439    466
STBY0     18     73    128    138
STBY1     18    243    298    308
STBY2     18    413    468    478
STOP0     83     87     91     99#
STOP1    253    257    261    269#
STOP2    423    427    431    439#
STOR0    102    115    121#   125
STOR1    272    285    291#   295
STOR2    442    455    461#   465
STORE    122    154    292    324    462    494    623#
SYSTAT    17    579    581    602    604
TOP       17    561
TTYBUF    17     38    106    117    208    276    287    378    446    457
UPDATE   596    610#
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                PAGE   2

USER      14
USER0     15      30#
USER1     15     200#
USER2     15     370#
VECFLG    18     582
ZERO1     18     156     326     496

CROSS REFERENCE COMPLETE
```

7.6 NOVAST AND NOVA

Modules NOVAST and NOVA are so intimately related that they must be discussed together. Basically, NOVAST is called as a subroutine of OPER about 120 times per second. The main function of NOVAST is to send characters to the Nova. Module NOVA is called as a subroutine from TB534 whenever a character arrives on USART3. Thus, module NOVA is responsible for interpreting the messages, prompts, and echos from the Nova.

Several variables are used by these two routines to coordinate their efforts. The main variables are as follows:

NOVSTS -- tells which of six routines in the NOVAST module should be executed the next time NOVAST gets control of the computer. The six main routines in NOVAST are: 0) STBCHK; 1) TIMTST; 2) CWECHO; 3) CHAOUT; 4) CRCHA; and 5) TENSEC.

NOINST -- tells which of six routines in the NOVA module should be executed the next time NOVA gets control of the computer. The six main routines in NOVA are 0) NOVTST; 1) NOVECH; 2) NOVSGN; 3) NOVXMT; 4) NVECH2; and 5) NOVBYE.

CHAPTR -- a pointer containing the address of the character being sent to the Nova.

CHAAMT -- the number of characters yet to be sent to the Nova.

ROUPTR -- the address of the next routine to be executed.

RUNFLG -- a flag which (when set to FF) indicates that the system is transmitting the command RUN"MICROINPUT) to the Nova.

When the Microprocessor Buffer System is first booted, the interrupt for the Nova is masked, therefore, module NOVA is not entered. NOVSTS is initialized at 0, therefore, when module NOVAST is called, the standby check routine (STBCHK) is executed.

STBCHK first checks the variable LSTUSR to see which of the three users was the last to have a report transmitted to the Nova. (LSTUSR is initialized as User 0 by SETUP.) STBCHK then checks whether there is a report standing by for any user, and the most recently serviced user is the last one checked. This rotating priority ensures that all three users get prompt attention. If the number of reports received for a user (STBYn) is different from the number of reports sent for that user (SENTn), STBCHK prepares to send a report to the NOVA. Otherwise, it simply returns control to OPER.

STBCHK performs several steps in preparing to send a report to the NOVA. First it sets the data terminal ready bit in USART3 by writing a 27_H to I/O address 17_H. To the NOVA, this is the same as a terminal being turned on. ROUPTR, CHAPTR, and CHAAMT are initialized at 0. The timer is sent for 10 seconds, as described below. The interrupt for the NOVA is unmasked. NOVSTS is set to 1, and NOINST is set to 0. An escape character is sent to the NOVA to get the attention of that computer, and finally control is returned to OPER.

There are six possible responses which the NOVA may give when the escape character arrives from the Microprocessor Buffer System: 1) ALL LINES BUSY, 2) MAX USERS, 3) asterisk, 4) LATS READY, 5) ACCOUNT ID, or 6) no response at all. If any response comes from the NOVA, the arriving characters cause interrupts. Because NOINST is 1, control of the computer is transferred to the NOVTST routine in module NOVA when these interrupts occur. If no response at all comes from the NOVA within ten seconds, the timer runs out and causes the Microprocessor Buffer System to start over.

The timer is routine TIMTST in module NOVAST. While the system is waiting for the NOVA to respond, module NOVAST still gets control of the computer 120 times per seond, and because NOVSTS is set to 1, TIMTST is executed each time. TIMSTS simply decrements a counter each time it is executed. The counter TIMCT1 is normally started at $1C00_H$, and (being decremented 120 times per second) reaches 0 at one minute. The counter TIMCTR is decremented whenever TIMCT1 reaches 0 to count the passing minutes. If the allowable delay elapses before routine NOVTST receives any of the five possible valid responses from the NOVA, TIMTST sends another escape character to the NOVA, and forces the system to start over in trying to transmit a report. TIMTST also sets the "NOVA not responding" error bit in the System Status Word (SYSTAT) so that this error is displayed on the console during the next console update.

If characters do arrive from the NOVA in response to the escape character, it is the responsibility of routine NOVTST in module NOVA to interpret them and act appropriately. IF NOVTST detects the messages ALL LINES BUSY or MAX USERS, it simply sets the timer for two minutes and relinquishes control of the computer. Consequently, when the two minutes have elapsed, the system will try again to get the NOVA's attention.

If the NOVA's response to the escape is an asterisk, the indication is that the Microprocessor Buffer System has acquired a NOVA port which was left active by some other user. Therefore, NOVTST sets ROUPTR to point to the NOVESC routine, sets CHAPTR to point to the string BYE), sets NOINST to 1, and sets NOVSTS to 2. The net result of all this is that the Microprocessor Buffer System will send the command BYE) to the NOVA and then send another escape character to try again to log in, with NOINST set to 0 to intercept the NOVA's response.

If the NOVA's response to the original escape is a LATS READY message, the system knows that it has acquired a NOVA port which was left waiting for an answer to the ACCOUNT ID prompt. Therefore, NOVTST simply sends another escape character to the NOVA to get another ACCOUNT ID prompt.

All the procedures above are unlikely; most often when the system sends the escape character to the NOVA, the NOVA immediately responds with the ACCOUNT ID prompt. In this case, NOVTST makes CHAPTR point to the proper user's account ID, sets NOVSTS to 3, and sets NOINST to 2.

NOVSTS having a value of 3 means that the next time NOVAST gets control, it executes its CHAOUT routine which sends a character to the NOVA and does not expect the NOVA to echo the character. CHAOUT first checks UART3 to see if it is ready to transmit. This is done by reading the status byte of UART3 from I/O address 17_H and determining whether the transmit ready (TXRDY) bit is set. If it is not, CHAOUT relinquishes control of the computer and checks the bit again 1/120 of a second later. When the UART is ready to transmit, CHAOUT gets the character pointed to by CHAPTR -- in this case, the first letter of the account ID. It sets the parity bit of that character, and sends it to the NOVA by writing it to I/O address 16_H. CHAOUT then increments CHAPTR and relinquishes control. 1/120 of a second later, CHAOUT again gets control, checks the UART, and sends the next character of the account ID to the NOVA. The last character of every account ID is a carriage return. When CHAOUT sends the carriage return to the NOVA, it sets the timer for five minutes, and sets NOVSTS to 1. Thus, 1/120 of a second later, the timer begins to decrement, and CHAOUT no longer gets control. NOINST remains set at 2.

Because NOINST is set at 2, when the NOVA responds to the account ID, the resulting interrupts pass control to routine NOVSGN of module NOVA. If the NOVA does not respond to the account ID within five minutes, the timer runs out, and the Microprocessor Buffer System sends another escape character to the NOVA. Normally, however, the NOVA does respond, and NOVSGN checks each character of the NOVA's response. NOVSGN looks for an asterick or a letter U in the NOVA's response: the date and time message sent by the NOVA is ignored. If NOVSGN finds a letter U in the NOVA response to the account ID, it assumes that the NOVA has sent one of the messages UNKNOWN ID or ID IN USE. In this case, NOVSGN masks the NOVA interrupt, sets the ID IN USE error bit in the System Status Word (SYSTAT), an sets the timer for two minutes so that the system will try again later when the ID IN USE error may be resolved. Obviously, an UNKNOWN ID error will never resolve itself.

If NOVSGN finds an asterisk in the response from the NOVA, it assumes that it has achieved control of the NOVA at BASIC command level. It, therefore, sets CHAPTR to point to the command RUN"MICROINPUT, sets NOINST to 1, sets NOVSTS to 2, and sets RUNFLG to FF_H to indicate that the string currently being sent to the NOVA is RUN"MICROINPUT.

With NOVSTS set to 2, the routine CWECHO of module NOVAST gets control of the computer 1/120 of a second later. CWECHO is the routine which sends to the NOVA a character for which an echo is expected. CWECHO first checks whether the UART for the NOVA is ready, and if it is not, CWECHO waits until it gets control the next time before proceeding. CWECHO then gets control the next time before proceeding. CWECHO then gets the character at CHAPTR, sets its parity bit, and sends it to the NOVA. It sets the timer for 20 minutes and sets NOVSTS to 1, so that the timer routine gets control on the next call to NOVAST. NOINST remains set at 1.

Because NOINST is 1, control is transferred to routine NOVECH of module NOVA when the echo of the character causes an interrupt. NOVECH increments CHAPTR to point to the next character of the string RUN"MICROINPUT) and sets NOVSTS to 2 so that CWECHO will send the next character to the NOVA the next time it gets control.

CWECHO keeps sending characters, and NOVECH keeps fielding their echos until CWECHO notices that it has sent the carriage return at the end of RUN"MICROINPUT). Then CWECHO sets NOINST to 3. It searches memory to find the first block of the report which caused this entire process to begin. It sets CHAAMT to the number of characters in the first block of the report, and it sets CHAPTR to point to the first character of the report. NOVSTS is set to 1.

With NOVSTS set to 1, the TIMTST routine begins counting down the time while the system waits for the NOVA to respond to the command RUN"MICROINPUT). The BASIC program MICROINPUT begins to run in the NOVA, and it sends a question mark prompt to the Microprocessor Buffer System when it is ready to receive a line of the report. Since NOINST is set to 3, control is transferred to the NOVXMT routine of the NOVA module when that question mark causes an interrupt.

NOVXMT gets the character at CHAPTR and examines it. If the character is a NUL, ENQ, LF, DC3, ESC, quotation mark, comma, backslash, or rubout, NOVXMT simply increments CHAPTR and decrements CHAAMT and goes on to the next character. If the first character of the line is a space, NOVXMT changes CHAPTR so that it points to the string <32>, which is the decimal equivalent of a space, in the form which can be recognized and not truncated by the NOVA. Normally, however, NOVXMT simply sets NOVSTS to 2 and NOINST to 4.

The next time NOVAST is called, the value of 2 for NOVSTS directs control to the routine CWECHO which gets the character at CHAPTR, sets its parity bit, sends it to the NOVA, and sets the timer for 20 minutes and NOVSTS to 1. When the echo of the character returns from the NOVA, the value of 4 for NOINST directs control to the routine NVECH2 of module NOVA. NVECH2 checks whether the echo matches the character that was sent. If it does not, NVECH2 sends a delete character to the NOVA. Usually, however, the echo matches, and NVECH2 increments CHAPTR and decrements CHAAMT. The NVECH2 checks the next character of the report. If the next character is a NUL, ENQ, LF, ESC, backslash, or delete, NVECH2 increments CHAPTR and decrements CHAAMT to skip the character. If the character is a comma, NVECH2 changes CHAPTR to point to the string <44>, which is the decimal equivalent of a comma and which is acceptable as part of a string in the NOVA. Finally, NVECH2 sets NOINST to 4 and NOVSTS to 2.

Thus, CWECHO keeps sending characters to the NOVA, and NVECH2 keeps fielding the echos of those characters until NVECH2 discovers that the next character to be sent is the carriage return at the end of a line of the report. For a carriage return, NVECH2 increments CHAPTR and decrements CHAAMT in advance, and then sets NOINST to 3 and NOVSTS to 2. When the program running in the NOVA gets the carriage return, it puts the line of the report into a disk file, and then it prompts for a new line by sending another question mark to the Microprocessor Buffer System. Because NOINST is set at 3, the interrupt triggered by the question mark transfers control to the routine NOVXMT, as described above.

Lines of the report are transmitted to the NOVA in this manner until NVECH2 finds that CHAAMT has reached 0. When this happens, NVECH2 checks whether or not there is more of the same report in another block. If there is, NVECH2 sets CHAPTR to point to the first character in that block, sets CHAAMT to the number of characters in that block, and continues. If there is no more to the report, NVECH2 sends an escape to the NOVA, sets NOINST to 5, and leaves the clock running with NOVSTS set at 1.

When the escape character is received by the program running in the NOVA, it knows that it has received the complete report. It processes the report into whatever files it is designed to create, and then lets the NOVA revert to BASIC command level. BASIC sends its asterisk prompt to the the Microprocessor Buffer System. Because NOINST is 5, the interrupt caused by the asterisk transfers control to routine NOVBYE in module NOVA. NOVBYE zeros the blocks where the report was in the main memory of the microcomputer. NOVBYE sets CHAPTR to point to the string BYE), sets ROUPTR to point the EXIT, sets NOVSTS to 2, and sets NOINST to 1.

With NOVSTS set to 2, CWECHO begins to send the string BYE to the NOVA, and since NOINST is set to 1, the echos of the characters cause NOVECH to be executed. When NOVECH detects the echo of the carriage return at the end of BYE), it jumps to the routine at ROUPTR, namely EXIT. EXIT sets the timer for ten seconds and sets NOVSTS to 5. EXIT also masks the NOVA interrupt and turns off the Data Terminal Ready bit in the UART for the NOVA.

Then, since the NOVA interrupt is masked, the module NOVA can no longer be entered. With NOVSTS set to 5, the routine TENSEC in module NOVAST gets control 120 times per second. TENSEC works similarily to the timer routine TIMTST. When the counter has reached zero, TENSEC changes NOVSTS to 0. This leaves the Microprocessor Buffer System in its original state, checking whether there is another report to be sent to the NOVA.

BEGIN FIRST LINE OF TEXT HERE

TYPING GUIDE SHEET

CENTER OF PAGE

TOP OF IMAGE AREA

DROPPED HEAD BEGIN SECTIONS HERE

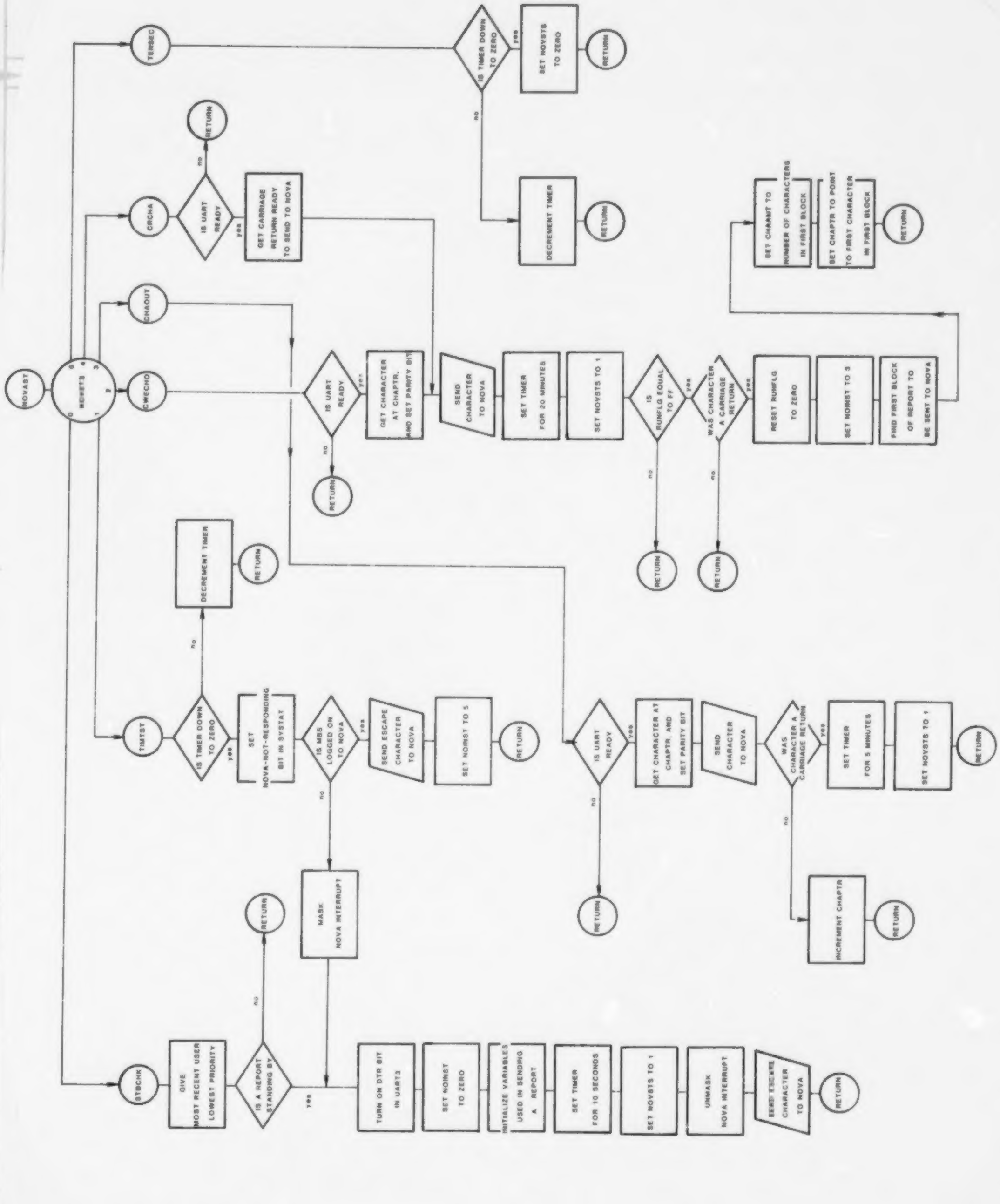

NOVAST
NOVSTS
0
1
2
3
4
TENSEC
IS TIMER DOWN TO ZERO
yes
no
SET NOVSTS TO ZERO
RETURN
DECREMENT TIMER
RETURN
CRCHA
IS UART READY
no
RETURN
yes
GET CARRIAGE RETURN READY TO SEND TO NOVA
CHAOUT
CWECHO
IS UART READY
yes
no
RETURN
GET CHARACTER AT CHAPTR, AND SET PARITY BIT
SEND CHARACTER TO NOVA
SET TIMER FOR 20 MINUTES
SET NOVSTS TO 1
IS RUNFLG EQUAL TO FF
yes
no
RETURN
WAS CHARACTER A CARRIAGE RETURN
yes
no
RETURN
RESET RUNFLG TO ZERO
SET NOINST TO 3
FIND FIRST BLOCK OF REPORT TO BE SENT TO NOVA
SET CHAMIT TO NUMBER OF CHARACTERS IN FIRST BLOCK
SET CHAPTR TO POINT TO FIRST CHARACTER IN FIRST BLOCK
RETURN
TIMTST
IS TIMER DOWN TO ZERO
no
DECREMENT TIMER
RETURN
yes
SET NOVA-NOT-RESPONDING BIT IN SYSTAT
IS MBS LOGGED ON TO NOVA
yes
SEND ESCAPE CHARACTER TO NOVA
SET NOINST TO 5
RETURN
no
MASK NOVA INTERRUPT
IS UART READY
no
RETURN
yes
GET CHARACTER AT CHAPTR, AND SET PARITY BIT
SEND CHARACTER TO NOVA
WAS CHARACTER A CARRIAGE RETURN
yes
SET TIMER FOR 5 MINUTES
SET NOVSTS TO 1
RETURN
no
INCREMENT CHAPTR
RETURN
STBCHK
GIVE MOST RECENT USER LOWEST PRIORITY
IS A REPORT STANDING BY
no
RETURN
yes
TURN ON DTR BIT IN UART3
SET NOINST TO ZERO
INITIALIZE VARIABLES USED IN SENDING A REPORT
SET TIMER FOR 10 SECONDS
SET NOVSTS TO 1
UNMASK NOVA INTERRUPT
SEND ESCAPE CHARACTER TO NOVA
RETURN

FIGURE 19. NOVAST FLOWCHART.

BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

BEGIN LAST LINE OF TEXT

3/8"

PAGE NUMBER

EPA-287 (Cin.)
(4-76)

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD: BEGIN SECTIONS HERE
BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"

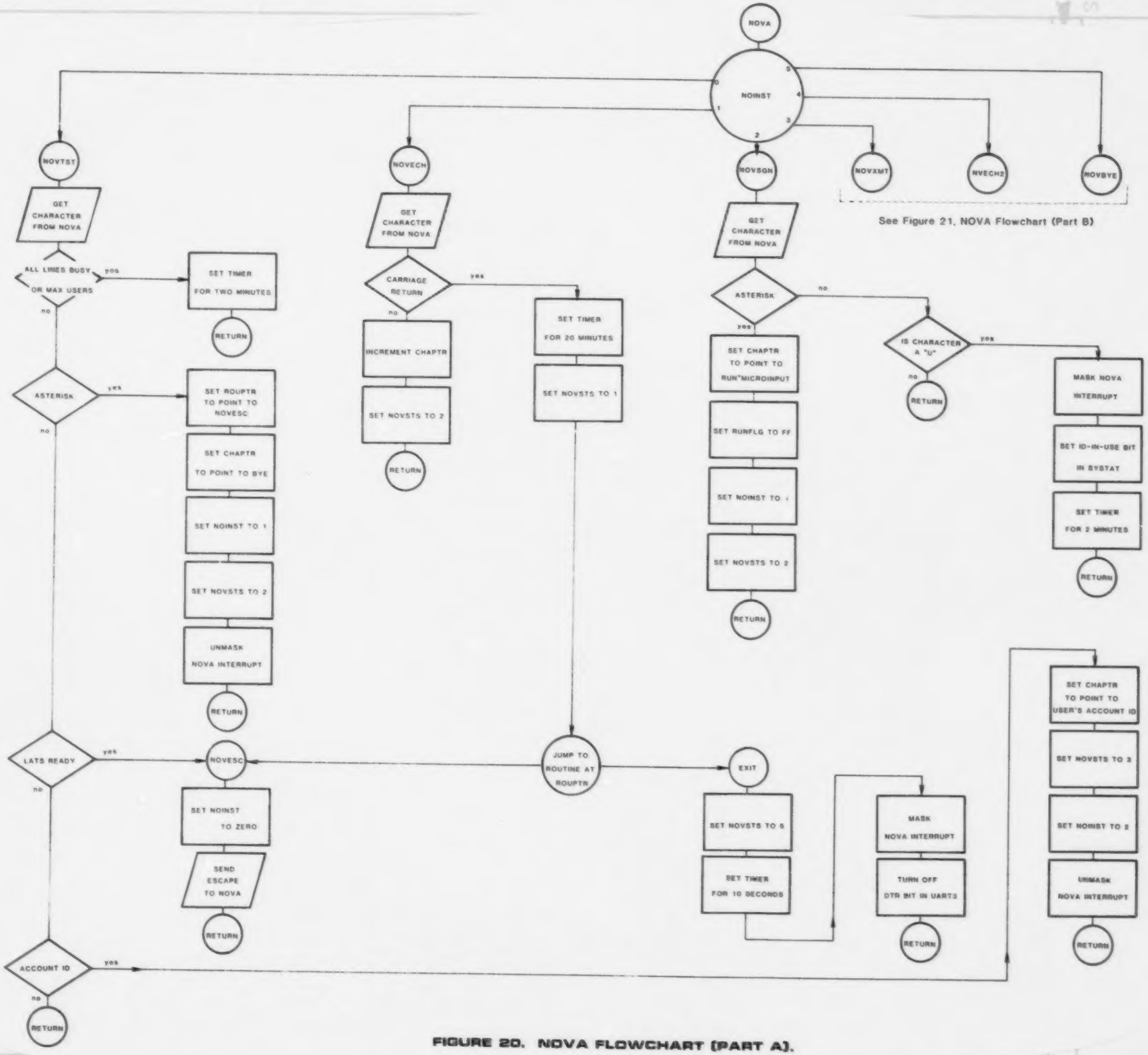

NOVA
NOINST
0
1
2
3
4
5
NOVTST
NOVECH
NOVSGN
NOVXMT
NVECH2
NOVBYE
See Figure 21, NOVA Flowchart (Part B)
GET CHARACTER FROM NOVA
ALL LINES BUSY OR MAX USERS
yes
no
SET TIMER FOR TWO MINUTES
RETURN
ASTERISK
SET ROUPTR TO POINT TO NOVESC
SET CHAPTR TO POINT TO BYE
SET NOINST TO 1
SET NOVSTS TO 2
UNMASK NOVA INTERRUPT
LATS READY
NOVESC
SET NOINST TO ZERO
SEND ESCAPE TO NOVA
ACCOUNT ID
CARRIAGE RETURN
INCREMENT CHAPTR
SET TIMER FOR 20 MINUTES
SET NOVSTS TO 1
JUMP TO ROUTINE AT ROUPTR
SET CHAPTR TO POINT TO RUN"MICROINPUT
SET RUNFLG TO FF
SET NOINST TO 1
IS CHARACTER A "U"
MASK NOVA INTERRUPT
SET ID-IN-USE BIT IN SYSTAT
SET TIMER FOR 2 MINUTES
EXIT
SET NOVSTS TO 5
SET TIMER FOR 10 SECONDS
MASK NOVA INTERRUPT
TURN OFF DTR BIT IN UART3
SET CHAPTR TO POINT TO USER'S ACCOUNT ID
SET NOVSTS TO 3
SET NOINST TO 2
UNMASK NOVA INTERRUPT

FIGURE 20. NOVA FLOWCHART (PART A).

EPA-287 (Cin.)
(4-76)

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
DROPPED HEAD; BEGIN SECTIONS HERE
CENTER
TOP OF IMAGE AREA

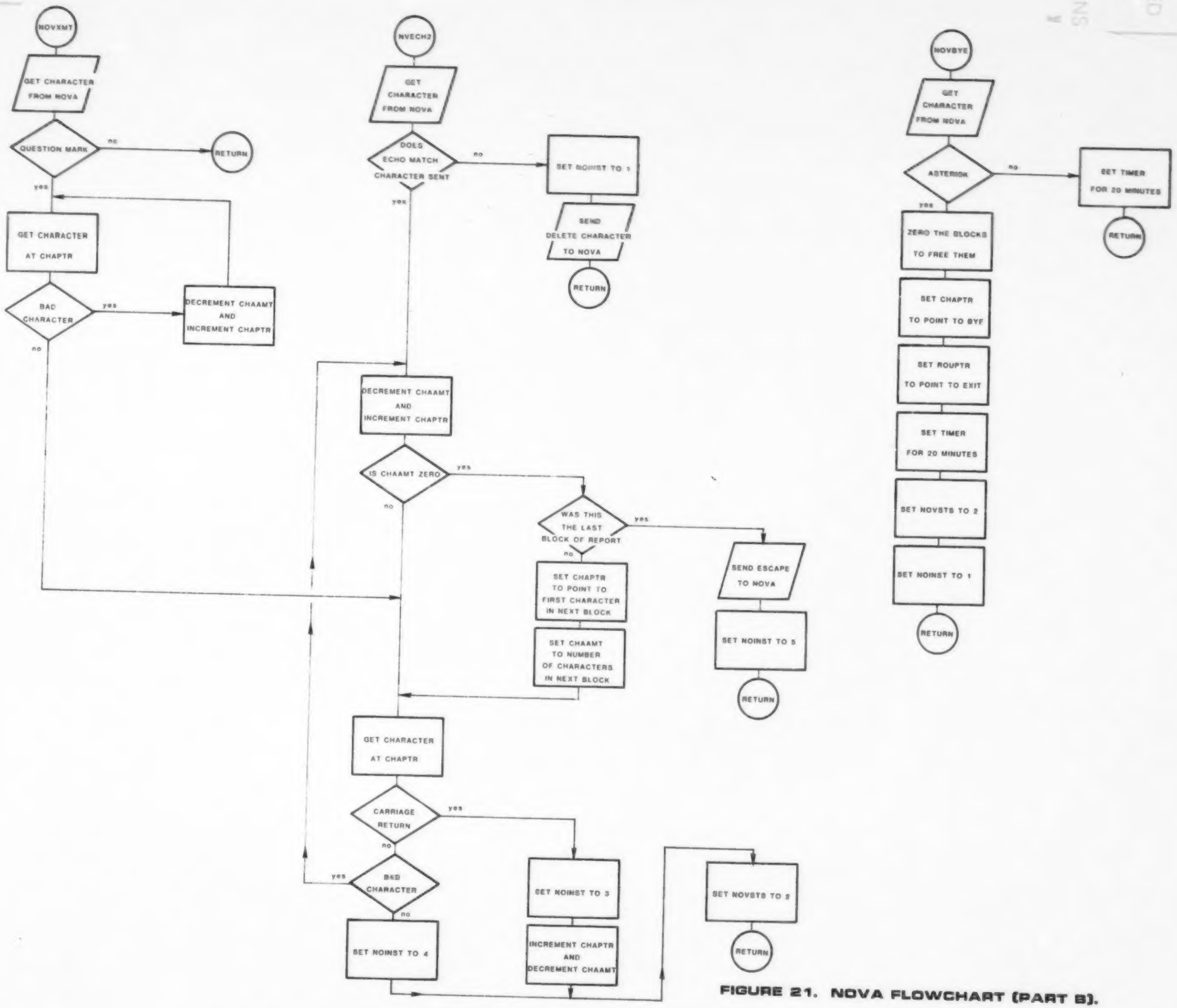

NOVXMT
GET CHARACTER FROM NOVA
QUESTION MARK
no
RETURN
yes
GET CHARACTER AT CHAPTR
BAD CHARACTER
yes
DECREMENT CHAAMT AND INCREMENT CHAPTR
no
NVECH2
GET CHARACTER FROM NOVA
DOES ECHO MATCH CHARACTER SENT
no
SET NOINST TO 1
SEND DELETE CHARACTER TO NOVA
RETURN
yes
DECREMENT CHAAMT AND INCREMENT CHAPTR
IS CHAAMT ZERO
yes
WAS THIS THE LAST BLOCK OF REPORT
yes
SEND ESCAPE TO NOVA
SET NOINST TO 5
RETURN
no
SET CHAPTR TO POINT TO FIRST CHARACTER IN NEXT BLOCK
SET CHAAMT TO NUMBER OF CHARACTERS IN NEXT BLOCK
no
GET CHARACTER AT CHAPTR
CARRIAGE RETURN
yes
SET NOINST TO 3
INCREMENT CHAPTR AND DECREMENT CHAAMT
no
BAD CHARACTER
yes
no
SET NOINST TO 4
SET NOVSTS TO 2
RETURN
NOVBYE
GET CHARACTER FROM NOVA
ASTERISK
no
SET TIMER FOR 20 MINUTES
RETURN
yes
ZERO THE BLOCKS TO FREE THEM
SET CHAPTR TO POINT TO BYF
SET ROUPTR TO POINT TO EXIT
SET TIMER FOR 20 MINUTES
SET NOVSTS TO 2
SET NOINST TO 1
RETURN

FIGURE 21. NOVA FLOWCHART (PART B).

BEGIN LAST LINE OF TEXT
3/8"
BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

```
ASM80 :F1:NOVAST.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    1


  LOC  OBJ          LINE        SOURCE STATEMENT

                       1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                       2 ;
                       3 ;         MODULE NAME:  NOVAST
                       4 ;
                       5 ;         DESCRIPTION:  THIS MODULE SENDS CHARACTERS FROM THE
                       6 ;                       MICROCOMPUTER TO THE NOVA.  THIS MODULE
                       7 ;                       ASSUMES CONTROL OF THE MICROCOMPUTER
                       8 ;                       ONCE AFTER EACH CHARACTER IS PRINTED ON THE CONSOLE
                       9 ;                       BY THE CONSOLE UPDATE MODULE.
                      10 ;
                      11 ;         MOST RECENT REVISION:  AUGUST 22, 1980
                      12 ;
                      13 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      14 ;
                      15         NAME    NOVAST
                      16         EXTRN   BOTTOM,CHAAMT,CHAPTR,CURBLK,FSTBLK,LSTUSR,NOINST,NOVSTS,ROUPTR,RUNF
                           LG
                      17         EXTRN   SNTP1,STBY0,SYSTAT,TIMCTR,TIMCT1,USRFLG
                      18         PUBLIC  ESCAPE,NOVAST,RETPAR,TIMINT
                      19         DSEG
                      20 ;
                      21 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      22 ;
                      23 ;     ROUTINE NAME:  NOVAST
                      24 ;     DESCRIPTION:  THIS ROUTINE EXAMINES THE FLAG "NOVSTS" AND JUMPS
                      25 ;                   TO THE APPROPRIATE ONE OF SIX OTHER MODULES.
                      26 ;
                      27 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      28 ;
 0000 F5              29 NOVAST: PUSH    PSW             ;SAVE A-REG AND FLAGS IN STACK
 0001 E5              30         PUSH    H               ;SAVE H-L-REG IN STACK
 0002 D5              31         PUSH    D               ;SAVE D-E-REG IN STACK
 0003 C5              32         PUSH    B               ;SAVE B-C-REG IN STACK
 0004 210000    E     33         LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 0007 7E              34         MOV     A,M             ;MOVE NOVSTS INTO A-REG
 0008 FE00            35         CPI     00H             ;IS NOVSTS ZERO?
 000A CA2900    D     36         JZ      STBCHK          ;IF YES, JUMP TO STBCHK
 000D FE01            37         CPI     01H             ;IS NOVSTS 1?
 000F CAB200    D     38         JZ      TIMTST          ;IF YES, JUMP TO TIMTST
 0012 FE02            39         CPI     02H             ;IS NOVSTS 2?
 0014 CA0E01    D     40         JZ      CWECHO          ;IS YES, JUMP TO CWECHO
 0017 FE03            41         CPI     03H             ;IS NOVSTS 3?
 0019 CAA401    D     42         JZ      CHAOUT          ;IF YES, JUMP TO CHAOUT
 001C FE04            43         CPI     04H             ;IS NOVSTS 4?
 001E CAD501    D     44         JZ      CRCHA           ;IF YES, JUMP TO CRCHA
 0021 FE05            45         CPI     05H             ;IS NOVSTS 5?
 0023 CAE301    D     46         JZ      TENSEC          ;IF YES, JUMP TO TENSEC
 0026 C30E02    D     47         JMP     RETPAR          ;IF NONE OF THE ABOVE, JUMP TO RETPAR
                      48 ;
                      49 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      50 ;
                      51 ;     ROUTINE NAME:  STBCHK
                      52 ;     DESCRIPTION:  THIS ROUTINE CHECKS TO SEE IF THERE IS ANY
                      53 ;                   COMPLETE REPORT READY TO BE TRANSMITTED TO THE
                      54 ;                   NOVA.  IF THERE IS, IT SENDS AN ESCAPE CHARACTER
                      55 ;                   TO THE  NOVA.  OTHERWISE IT RELINQUISHES CONTROL OF
                      56 ;                   THE MICROCOMPUTER.
                      57 ;
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    2

  LOC  OBJ         LINE        SOURCE STATEMENT

                     58 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     59 ;
  0029 210000   E    60 STBCHK: LXI    H,LSTUSR        ;LOAD ADDRESS OF LSTUSR INTO H-L-REG
  002C 7E            61         MOV    A,M             ;MOVE LSTUSR INTO A-REG
  002D 3C            62 STB1:   INR    A               ;INCREMENT LSTUSR TO GET USRFLG IN A-REG
  002E FE03          63         CPI    03H             ;IS USRFLG UP TO 3?
  0030 C23500   D    64         JNZ    STB2            ;IF NO, JUMP TO STB2
  0033 3E00          65         MVI    A,00H           ;IF YES, SET USRFLG BACK TO ZERO
  0035 210000   E    66 STB2:   LXI    H,STBY0         ;LOAD ADDRESS OF STBY0 INTO H-L-REG
  0038 FE00          67         CPI    00H             ;IS USRFLG ZERO?
  003A CA4400   D    68         JZ     STB3            ;IF YES, JUMP TO STB3
  003D 24            69         INR    H               ;INCREMENT H REG TO PRODUCE ADDRESS OF STBY1
  003E FE01          70         CPI    01H             ;IS USRFLG 1?
  0040 CA4400   D    71         JZ     STB3            ;IF YES, JUMP TO STB3
  0043 24            72         INR    H               ;INCREMENT H REG TO PRODUCE ADDRESS OF STBY2
  0044 F5            73 STB3:   PUSH   PSW             ;STORE USRFLG IN STACK
  0045 7E            74         MOV    A,M             ;MOVE STBYn INTO A-REG
  0046 2E18          75         MVI    L,18H           ;CHANGE H-L-REG TO ADDRESS OF SENTn
  0048 BE            76         CMP    M               ;IS STBYn EQUAL TO SENTn?
  0049 C25700   D    77         JNZ    STB4            ;IF NO, JUMP TO STB4
  004C F1            78         POP    PSW             ;GET USRFLG BACK FROM STACK INTO A-REG
  004D 210000   E    79         LXI    H,LSTUSR        ;LOAD ADDRESS OF LSTUSR INTO H-L-REG
  0050 BE            80         CMP    M               ;IS USRFLG EQUAL TO LSTUSR?
  0051 C22D00   D    81         JNZ    STB1            ;IF NO, JUMP BACK TO STB1
  0054 C30E02   D    82         JMP    RETPAR          ;IF YES, JUMP TO RETPAR
  0057 F1            83 STB4:   POP    PSW             ;GET USRFLG BACK FROM STACK INTO A-REG
  0058 46            84         MOV    B,M             ;MOVE SENTn INTO B-REG
  0059 04            85         INR    B               ;INCREMENT SENTn
  005A 210000   E    86         LXI    H,SNTP1         ;LOAD ADDRESS OF SNTP1 INTO H-L-REG
  005D 70            87         MOV    M,B             ;UPDATE SNTP1
  005E 210000   E    88         LXI    H,LSTUSR        ;LOAD ADDRESS OF LSTUSR INTO H-L-REG
  0061 77            89         MOV    M,A             ;UPDATE LSTUSR
  0062 210000   E    90         LXI    H,USRFLG        ;LOAD ADDRESS OF USRFLG INTO H-L-REG
  0065 77            91         MOV    M,A             ;UPDATE USRFLG
  0066 3E27          92 ESCAPE: MVI    A,27H           ;MOVE USART COMMAND WORD INTO A-REG
  0068 D317          93         OUT    17H             ;PROGRAM USART-3 TO SET DATA TERMINAL READY BIT
  006A 3E00          94         MVI    A,00H           ;MOVE ZERO INTO A-REG
  006C 210000   E    95         LXI    H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
  006F 77            96         MOV    M,A             ;SET NOINST TO ZERO
  0070 210000   E    97         LXI    H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
  0073 77            98         MOV    M,A             ;SET ROUPTR-MSB TO ZERO
  0074 23            99         INX    H               ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
  0075 77           100         MOV    M,A             ;SET ROUPTR-LSB TO ZERO
  0076 210000   E   101         LXI    H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  0079 77           102         MOV    M,A             ;SET CHAPTR-MSB TO ZERO
  007A 23           103         INX    H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  007B 77           104         MOV    M,A             ;SET CHAPTR-LSB TO ZERO
  007C 210000   E   105         LXI    H,CHAAMT        ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
  007F 77           106         MOV    M,A             ;SET CHAAMT-MSB TO ZERO
  0080 23           107         INX    H               ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
  0081 77           108         MOV    M,A             ;SET CHAAMT-LSB TO ZERO
  0082 210000   E   109         LXI    H,TIMCTR        ;LOAD ADDRESS OF TIMCTR INTO H-L-REG
  0085 3601         110         MVI    M,01H           ;SET TIMCTR TO 1
  0087 210000   E   111         LXI    H,TIMCT1        ;LOAD ADDRESS OF TIMCT1-MSB INTO H-L-REG
  008A 3605         112         MVI    M,05H           ;SET TIMCT1-MSB TO 5
  008C 23           113         INX    H               ;INCREMENT H-L-REG TO ADDRESS OF TIMCT1-LSB
  008D 3600         114         MVI    M,00H           ;SET TIMCT1-LSB TO 0
  008F 210000   E   115         LXI    H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
  0092 3601         116         MVI    M,01H           ;SET NOVSTS TO 1
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    3

  LOC  OBJ          LINE        SOURCE STATEMENT

  0094 DB19          117           IN      19H               ;READ CURRENT INTERRUPT MASK REGISTER FROM PIC-0
  0096 E6BF          118           ANI     0BFH              ;RESET MASK BIT FOR NOVA INTERRUPT (INTERRUPT-6)
  0098 D319          119           OUT     19H               ;PROGRAM PIC-0 TO UNMASK NOVA INTERRUPT
  009A 3E1B          120           MVI     A,1BH             ;MOVE ESC CHARACTER INTO A-REG
  009C F680          121           ORI     80H               ;SET PARITY BIT
  009E D316          122           OUT     16H               ;SEND ESC TO NOVA THROUGH USART-3
  00A0 C30E02   D    123           JMP     RETPAR            ;JUMP TO RETPAR
                     124 ;
                     125 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     126 ;
                     127 ;     ROUTINE NAME:  TIMINT
                     128 ;     DESCRIPTION:  THIS ROUTINE INITIALIZES THE TIMER BY SETTING
                     129 ;                   TIMCT1 TO 1C00H AND SETTING TIMCTR TO THE NUMBER OF
                     130 ;                   MINUTES SPECIFIED IN THE ACCUMULATOR.
                     131 ;
                     132 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     133 ;
  00A3 E5            134 TIMINT: PUSH    H                 ;STORE H-L-REG IN STACK WHILE H-L-REG IS IN USE
  00A4 210000   E    135           LXI     H,TIMCT1          ;LOAD ADDRESS OF TIMCT1-MSB INTO H-L-REG
  00A7 361C          136           MVI     M,1CH             ;MOVE 1CH INTO TIMCT1-MSB
  00A9 23            137           INX     H                 ;INCREMENT H-L-REG TO ADDRESS OF TIMCT1-LSB
  00AA 3600          138           MVI     M,00H             ;MOVE 00H INTO TIMCT1-LSB
  00AC 210000   E    139           LXI     H,TIMCTR          ;MOVE ADDRESS OF TIMCTR INTO H-L-REG
  00AF 77            140           MOV     M,A               ;MOVE NUMBER OF MINUTES INTO TIMCTR
  00B0 E1            141           POP     H                 ;RESTORE CONTENTS OF H-L-REG AFTER USE
  00B1 C9            142           RET                       ;RETURN TO INVOKING PROCEDURE
                     143 ;
                     144 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     145 ;
                     146 ;     ROUTINE NAME:  TIMTST
                     147 ;     DESCRIPTION:  THIS ROUTINE CHECKS TO SEE IF THE TIME SPECIFIED IN
                     148 ;                   TIMCTR AND TIMCT1 HAS ELAPSED.  IF NOT, IT DECREMENTS
                     149 ;                   THE TIMER.  IF THE TIME HAS ELAPSED, THIS ROUTINE
                     150 ;                   TURNS ON THE "NOVA NOT RESPONDING" ERROR BIT AND SENDS
                     151 ;                   AN ESCAPE CHARACTER TO THE NOVA.
                     152 ;
                     153 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     154 ;
  00B2 1600          155 TIMTST: MVI     D,00H             ;INITIALIZE D-REG AT ZERO
  00B4 210000   E    156           LXI     H,TIMCT1          ;LOAD ADDRESS OF TIMCT1-MSB INTO H-L-REG
  00B7 7E            157           MOV     A,M               ;MOVE TIMCT1-MSB INTO A-REG
  00B8 47            158           MOV     B,A               ;MOVE TIMCT1-MSB INTO B-REG
  00B9 FE00          159           CPI     00H               ;IS TIMCT1-MSG ZERO?
  00BB C2BF00   D    160           JNZ     TIM1              ;IF NO, JUMP TO TIM1
  00BE 14            161           INR     D                 ;IF YES, INCREMENT D
  00BF 23            162 TIM1:     INX     H                 ;INCREMENT H-L-REG TO ADDRESS OF TIMCT1-LSB
  00C0 7E            163           MOV     A,M               ;MOVE TIMCT1-LSB INTO A-REG
  00C1 4F            164           MOV     C,A               ;MOVE TIMCT1-LSB INTO C-REG
  00C2 FE00          165           CPI     00H               ;IS TIMCT1-LSB ZERO?
  00C4 C2C800   D    166           JNZ     TIM2              ;IF NO, JUMP TO TIM2
  00C7 14            167           INR     D                 ;IF YES, INCREMENT D-REG
  00C8 7A            168 TIM2:     MOV     A,D               ;MOVE COUNTER FROM D-REG INTO A-REG
  00C9 FE02          169           CPI     02H               ;IS COUNTER 2?
  00CB CAD500   D    170           JZ      TIM3              ;IF YES, THEN JUMP TO TIM3
  00CE 0B            171           DCX     B                 ;IF NO, DECREMENT TIMCT1 IN B-C-REG
  00CF 71            172           MOV     M,C               ;MOVE TIMCT1-LSB BACK TO MEMORY
  00D0 2B            173           DCX     H                 ;DECREMENT H-L-REG BACK TO ADDRESS OF TIMCT1-MSB
  00D1 70            174           MOV     M,B               ;MOVE TIMCT1-MSB BACK TO MEMORY
  00D2 C30E02   D    175           JMP     RETPAR            ;JUMP TO RETPAR
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    4

  LOC  OBJ         LINE        SOURCE STATEMENT

 00D5 210000   E    176 TIM3:   LXI     H,TIMCTR           ;LOAD ADDRESS OF TIMCTR INTO H-L-REG
 00D8 7E            177         MOV     A,M                ;MOVE TIMCTR INTO A-REG
 00D9 FE01          178         CPI     01H                ;IS TIMCTR DOWN TO 1?
 00DB CAE500   D    179         JZ      TIM4               ;IF YES, JUMP TO TIM4
 00DE 3D            180         DCR     A                  ;IF NO, DECREMENT TIMCTR IN A-REG
 00DF CDA300   D    181         CALL    TIMINT             ;RESET TIMER
 00E2 C30E02   D    182         JMP     RETPAR             ;JUMP TO RETPAR
 00E5 210000   E    183 TIM4:   LXI     H,SYSTAT           ;LOAD ADDRESS OF SYSTAT INTO H-L-REG
 00E8 7E            184         MOV     A,M                ;MOVE SYSTAT INTO A-REG
 00E9 E601          185         ANI     01H                ;REMOVE ALL ERROR BITS EXCEPT MEMORY OVERFLOW
 00EB F608          186         ORI     08H                ;SET BIT FOR NOVA NOT RESPONDING ERROR
 00ED 77            187         MOV     M,A                ;UPDATE SYSTAT IN MEMORY
 00EE 210000   E    188         LXI     H,USRFLG           ;LOAD ADDRESS OF USRFLG INTO H-L-REG
 00F1 7E            189         MOV     A,M                ;MOVE USRFLG INTO A-REG
 00F2 FEFF          190         CPI     0FFH               ;IS USRFLG FF?
 00F4 CA0001   D    191         JZ      TIM5               ;IF YES, JUMP TO TIM5
 00F7 DB19          192         IN      19H                ;READ CURRENT INTERRUPT MASK REGISTER FROM PIC-0
 00F9 F680          193         ORI     80H                ;SET MASK BIT FOR NOVA INTERRUPT (INTERRUPT-6)
 00FB D319          194         OUT     19H                ;PROGRAM PIC-0 TO MASK NOVA INTERRUPT
 00FD C36600   D    195         JMP     ESCAPE             ;JUMP TO ESCAPE
 0100 3E1B          196 TIM5:   MVI     A,1BH              ;MOVE ESCAPE CHARACTER INTO A-REG
 0102 F680          197         ORI     80H                ;SET PARITY BIT
 0104 D316          198         OUT     16H                ;SEND ESCAPE CHARACTER TO NOVA THROUGH USART-3
 0106 210000   E    199         LXI     H,NOINST           ;LOAD ADDRESS OF NOINST INTO H-L-REG
 0109 3605          200         MVI     M,05H              ;SET NOINST TO 5
 010B C30E02   D    201         JMP     RETPAR             ;JUMP TO RETPAR
                    202 ;
                    203 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    204 ;
                    205 ;       ROUTINE NAME:  CWECHO
                    206 ;       DESCRIPTION:   THIS ROUTINE SENDS TO THE NOVA ONE CHARACTER FOR
                    207 ;                      WHICH AN ECHO IS EXPECTED.  WHEN THIS ROUTINE
                    208 ;                      SENDS THE CARRIAGE RETURN AFTER 'RUN"MICROINPUT',
                    209 ;                      IT ALSO LOCATES THE BEGINNING OF THE REPORT
                    210 ;                      TO BE TRANSMITTED AND SETS THE VARIABLES NECESSARY
                    211 ;                      TO PERFORM THAT OPERATION.
                    212 ;
                    213 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    214 ;
 010E DB17          215 CWECHO: IN      17H                ;READ STATUS BYTE OF USART-3
 0110 E601          216         ANI     01H                ;ELIMINATE ALL BITS BUT TXRDY
 0112 FE01          217         CPI     01H                ;IS TXRDY SET?
 0114 C20E02   D    218         JNZ     RETPAR             ;IF NO, JUMP TO RETPAR
 0117 210000   E    219         LXI     H,CHAPTR           ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 011A 56            220         MOV     D,M                ;MOVE CHAPTR-MSB INTO D-REG
 011B 23            221         INX     H                  ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 011C 5E            222         MOV     E,M                ;MOVE CHAPTR-LSB INTO E-REG
 011D EB            223         XCHG                       ;SWAP CHAPTR INTO H-L-REG
 011E 7E            224         MOV     A,M                ;MOVE CHARACTER AT CHAPTR INTO A-REG
 011F 47            225         MOV     B,A                ;MOVE CHARACTER TO B-REG ALSO
 0120 F680          226 CWE1:   ORI     80H                ;SET PARITY BIT
 0122 D316          227         OUT     16H                ;SEND CHARACTER TO NOVA THROUGH USART-3
 0124 3E14          228         MVI     A,14H              ;SET TIME AT 20 MINUTES
 0126 CDA300   D    229         CALL    TIMINT             ;INITIALIZE TIMER
 0129 210000   E    230         LXI     H,NOVSTS           ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 012C 3601          231         MVI     M,01H              ;SET NOVSTS TO 1
 012E 3A0000   E    232         LDA     RUNFLG             ;LOAD RUNFLG INTO A-REG
 0131 FEFF          233         CPI     0FFH               ;IS RUNFLG SET TO FF?
 0133 C20E02   D    234         JNZ     RETPAR             ;IF NO, JUMP TO RETPAR
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    5

  LOC  OBJ          LINE        SOURCE STATEMENT

  0136 78            235        MOV     A,B                  ;MOVE CHARACTER BACK INTO A-REG
  0137 FE0D          236        CPI     0DH                  ;IS CHARACTER A CARRIAGE RETURN?
  0139 C20E02   D    237        JNZ     RETPAR               ;IF NO, JUMP TO RETPAR
  013C 210000   E    238        LXI     H,RUNFLG             ;LOAD ADDRESS OF RUNFLG INTO H-L-REG
  013F 3600          239        MVI     M,00H                ;RESET RUNFLG BACK TO ZERO
  0141 3E03          240        MVI     A,03H                ;MOV 3 INTO A-REG
  0143 320000   E    241        STA     NOINST               ;SET NOINST TO 3
  0146 210000   E    242        LXI     H,USRFLG             ;LOAD ADDRESS OF USRFLG INTO H-L-REG
  0149 7E            243        MOV     A,M                  ;MOVE USRFLG INTO A-REG
  014A 3C            244        INR     A                    ;CHANGE USRFLG FROM 0,1,2 TO 1,2,3 SCHEME
  014B 4F            245        MOV     C,A                  ;STORE USRFLG IN C-REG
  014C 210000   E    246        LXI     H,SNTP1              ;LOAD ADDRESS OF SNTP1 INTO H-L-REG
  014F 46            247        MOV     B,M                  ;STORE SNTP1 IN B-REG
  0150 210000   E    248        LXI     H,BOTTOM             ;LOAD ADDRESS OF BOTTOM INTO H-L-REG
  0153 79            249 MT1:   MOV     A,C                  ;MOVE USRFLG INTO A-REG
  0154 BE            250        CMP     M                    ;IS FIRST BYTE OF BLOCK EQUAL TO USRFLG?
  0155 C26701   D    251        JNZ     MT2                  ;IF NO, JUMP TO MT2
  0158 23            252        INX     H                    ;INCREMENT H-L-REG TO ADDRESS OF 2ND BYTE OF BLOCK
  0159 3EFF          253        MVI     A,0FFH               ;MOVE FF INTO A-REG
  015B BE            254        CMP     M                    ;IS 2ND BYTE OF BLOCK FF?
  015C C26701   D    255        JNZ     MT2                  ;IF NO, JUMP TO MT2
  015F 23            256        INX     H                    ;INCREMENT H-L-REG TO ADDRESS OF 5TH BYTE OF BLOCK
  0160 23            257        INX     H
  0161 23            258        INX     H
  0162 78            259        MOV     A,B                  ;MOVE SNTP1 INTO A-REG
  0163 BE            260        CMP     M                    ;IS 3RD BYTE OF BLOCK EQUAL TO SNTP1?
  0164 CA7001   D    261        JZ      MT3                  ;IF YES, JUMP TO MT3
  0167 7C            262 MT2:   MOV     A,H                  ;MOVE MSB OF BLOCK ADDRESS TO A-REG
  0168 C604          263        ADI     04H                  ;ADD 4 TO MSB OF BLOCK ADDRESS
  016A 67            264        MOV     H,A                  ;MOVE MSB OF BLOCK ADDRESS BACK TO H-REG
  016B 2E00          265        MVI     L,00H                ;ZERO L-REG
  016D C35301   D    266        JMP     MT1                  ;GO BACK TO CHECK NEXT BLOCK
  0170 E5            267 MT3:   PUSH    H                    ;STORE H-L-REG IN STACK
  0171 44            268        MOV     B,H                  ;MOVE MSB OF ADDRESS OF BLOCK INTO B-REG
  0172 210000   E    269        LXI     H,CURBLK             ;LOAD ADDRESS OF CURBLK INTO H-L-REG
  0175 70            270        MOV     M,B                  ;MOVE MSB OF ADDRESS OF BLOCK INTO CURBLK IN MEMORY
  0176 210000   E    271        LXI     H,FSTBLK             ;LOAD ADDRESS OF FSTBLK INTO H-L-REG
  0179 70            272        MOV     M,B                  ;MOVE MSB OF ADDRESS OF BLOCK INTO FSTBLK IN MEMORY
  017A E1            273        POP     H                    ;RESTORE CONTENTS OF H-L-REG
  017B 2B            274        DCX     H                    ;DECREMENT H-L-REG TO ADDRESS OF 3RD BYTE OF BLOCK
  017C 2B            275        DCX     H
  017D 5E            276        MOV     E,M                  ;MOVE CONTENTS OF 3RD BYTE (CHAAMT-LSB) TO E-REG
  017E 23            277        INX     H                    ;INCREMENT H-L-REG TO ADDRESS OF 4TH BYTE OF BLOCK
  017F 56            278        MOV     D,M                  ;MOVE CONTENTS OF 4TH BYTE (CHAAMT-MSB) TO D-REG
  0180 23            279        INX     H                    ;INCREMENT H-L-REG TO ADDRESS OF 1ST CHARACTER IN B
                         LOCK
  0181 23            280        INX     H
  0182 E5            281        PUSH    H                    ;STORE IN STACK THE ADDRESS OF FIRST CHARACTER
  0183 210000   E    282        LXI     H,CHAAMT             ;LOAD ADDRESS OF CHAAMT INTO H-L-REG
  0186 7A            283        MOV     A,D                  ;MOVE MSB INTO A-REG
  0187 FE03          284        CPI     03H                  ;IS MSB 3?
  0189 C29301   D    285        JNZ     MT4                  ;IF NO, JUMP TO MT4
  018C 7B            286        MOV     A,E                  ;MOVE LSB INTO A-REG
  018D FEFF          287        CPI     0FFH                 ;IS LSB FF?
  018F C29301   D    288        JNZ     MT4                  ;IF NO, JUMP TO MT4
  0192 1B            289        DCX     D                    ;DECREMENT D-E-REG TO AVOID COUNTING LAST CHARACTER
                          IN BLOCK
  0193 1B            290 MT4:   DCX     D                    ;SUBTRACT 4 FROM D-E-REG TO AVOID COUNTING FIRST CH
                         ARACTERS
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

0194 1B            291         DCX     D
0195 1B            292         DCX     D
0196 1B            293         DCX     D
0197 72            294         MOV     M,D             ;UPDATE CHAAMT-MSB IN MEMORY
0198 23            295         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
0199 73            296         MOV     M,E             ;UPDATE CHAAMT-LSB IN MEMORY
019A D1            297         POP     D               ;GET ADDRESS OF 1ST CHARACTER OF BLOCK INTO D-E-REG
019B 210000     E  298         LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
019E 72            299         MOV     M,D             ;UPDATE CHAPTR-MSB IN MEMORY
019F 23            300         INX     H               ;INCRMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
01A0 73            301         MOV     M,E             ;UPDATE CHAPTR-LSB IN MEMORY
01A1 C30E02     D  302         JMP     RETPAR          ;JUMP TO RETPAR
                   303 ;
                   304 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   305 ;
                   306 ;      ROUTINE NAME:  CHAOUT
                   307 ;      DESCRIPTION:  THIS ROUTINE SENDS TO THE NOVA ONE CHARACTER
                   308 ;                    FOR WHICH NO ECHO IS EXPECTED.
                   309 ;
                   310 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   311 ;
01A4 DB17          312 CHAOUT: IN      17H             ;READ STATUS BYTE OF USART-3
01A6 E601          313         ANI     01H             ;ELIMINATE ALL BITS BUT TXRDY
01A8 FE01          314         CPI     01H             ;IS TXRDY SET?
01AA C20E02     D  315         JNZ     RETPAR          ;IF NO, JUMP TO RETPAR
01AD 210000     E  316         LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
01B0 56            317         MOV     D,M             ;MOVE CHAPTR-MSB INTO D-REG
01B1 23            318         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
01B2 5E            319         MOV     E,M             ;MOVE CHAPTR-LSB INTO E-REG
01B3 EB            320         XCHG                    ;SWAP CHAPTR INTO H-L-REG
01B4 7E            321         MOV     A,M             ;MOVE CHARACTER AT CHAPTR INTO A-REG
01B5 F680          322         ORI     80H             ;SET PARITY BIT
01B7 D316          323         OUT     16H             ;SEND CHARACTER TO NOVA THROUGH USART-3
01B9 E67F          324         ANI     7FH             ;RESET PARITY BIT
01BB FE0D          325         CPI     0DH             ;IS CHARACTER A CARRIAGE RETURN?
01BD CAC801     D  326         JZ      CHA1            ;IF YES, JUMP TO CHA1
01C0 23            327         INX     H               ;INCREMENT CHAPTR IN H-L-REG
01C1 EB            328         XCHG                    ;SWAP CHAPTR BACK INTO D-E-REG
01C2 73            329         MOV     M,E             ;UPDATE CHAPTR-LSB
01C3 2B            330         DCX     H               ;DECREMENT H-L-REG BACK TO ADDRESS OF CHAPTR-MSB
01C4 72            331         MOV     M,D             ;UPDATE CHAPTR-MSB
01C5 C30E02     D  332         JMP     RETPAR          ;JUMP TO RETPAR
01C8 3E05          333 CHA1:   MVI     A,05H           ;SET TIME FOR 5 MINUTES
01CA CDA300     D  334         CALL    TIMINT          ;INITIALIZE TIMER
01CD 210000     E  335         LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
01D0 3601          336         MVI     M,01H           ;SET NOVSTS TO 1
01D2 C30E02     D  337         JMP     RETPAR          ;JUMP TO RETPAR
                   338 ;
                   339 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   340 ;
                   341 ;      ROUTINE NAME:  CRCHA
                   342 ;      DESCRIPTION:  THIS ROUTINE SETS UP A CARRIAGE RETURN FOR
                   343 ;                    TRANSMISSION TO THE NOVA.  SINCE THE SYSTEM
                   344 ;                    DOES NOT CHECK FOR THE ECHO OF A CARRIAGE
                   345 ;                    RETURN, THE VARIABLES CHAAMT AND CHAPTR MUST
                   346 ;                    BE HANDLED SPECIALLY IN THIS CASE.
                   347 ;
                   348 ;;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   349 ;
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    7

 LOC  OBJ         LINE        SOURCE STATEMENT

 01D5 DB17         350 CRCHA:  IN      17H             ;READ STATUS BYTE OF USART-3
 01D7 E601         351         ANI     01H             ;ELIMINATE ALL BITS BUT TXRDY
 01D9 FE01         352         CPI     01H             ;IS TXRDY SET?
 01DB C20E02   D   353         JNZ     RETPAR          ;IF NO, JUMP TO RETPAR
 01DE 3E0D         354         MVI     A,0DH           ;MOVE CARRIAGE RETURN INTO A-REG
 01E0 C32001   D   355         JMP     CWE1            ;JUMP TO CWE1
                   356 ;
                   357 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   358 ;
                   359 ;       ROUTINE NAME:  TENSEC
                   360 ;       DESCRIPTION:   THIS ROUTINE IS USED TO CAUSE A TEN-SECOND
                   361 ;                      DELAY BETWEEN THE TIME THE MICROCOMPUTER LOGS
                   362 ;                      OFF FROM THE NOVA AND THE NEXT TIME IT SENDS
                   363 ;                      AN ESCAPE TO TRY TO LOG ONTO THE NOVA.
                   364 ;
                   365 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   366 ;
 01E3 0600         367 TENSEC: MVI     B,00H           ;SET COUNTER AT ZERO IN B-REG
 01E5 210000   E   368         LXI     H,TIMCT1        ;LOAD ADDRESS OF TIMCT1-MSB INTO H-L-REG
 01E8 7E           369         MOV     A,M             ;MOVE TIMCT1-MSB INTO A-REG
 01E9 57           370         MOV     D,A             ;MOVE TIMCT1-MSB INTO D-REG ALSO
 01EA FE00         371         CPI     00H             ;IS TIMCT1-MSB A ZERO?
 01EC CAF001   D   372         JZ      TEN1            ;IF YES, JUMP TO TEN1
 01EF 04           373         INR     B               ;INCREMENT COUNTER IN B-REG
 01F0 23           374 TEN1:   INX     H               ;INCREMENT H-L-REG TO ADDRESS OF TIMCT1-LSB
 01F1 7E           375         MOV     A,M             ;MOVE TIMCT1-LSB TO A-REG
 01F2 5F           376         MOV     E,A             ;MOVE TIMCT1-LSB TO E-REG ALSO
 01F3 FE00         377         CPI     00H             ;IS TIMCT1-LSB A ZERO?
 01F5 CAF901   D   378         JZ      TEN2            ;IF YES, JUMP TO TEN2
 01F8 04           379         INR     B               ;INCREMENT COUNTER IN B-REG
 01F9 78           380 TEN2:   MOV     A,B             ;MOVE COUNTER INTO A-REG
 01FA FE00         381         CPI     00H             ;IS COUNTER STILL ZERO?
 01FC CA0602   D   382         JZ      TEN3            ;IF YES, JUMP TO TEN3
 01FF 1B           383         DCX     D               ;DECREMENT TIMCT1
 0200 73           384         MOV     M,E             ;UPDATE TIMCT1-LSB IN MEMORY
 0201 2B           385         DCX     H               ;DECREMENT H-L-REG TO ADDRESS OF TIMCT1-MSB
 0202 72           386         MOV     M,D             ;UPDATE TIMCT1-MSB IN MEMORY
 0203 C30E02   D   387         JMP     RETPAR          ;JUMP TO RETPAR
 0206 210000   E   388 TEN3:   LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 0209 3600         389         MVI     M,00H           ;SET NOVSTS TO 00
 020B C30E02   D   390         JMP     RETPAR          ;JUMP TO RETPAR
                   391 ;
                   392 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   393 ;
                   394 ;       ROUTINE NAME:  RETPAR
                   395 ;       DESCRIPTION:   THIS ROUTINE RESTORES THE REGISTERS TO THEIR
                   396 ;                      ORIGINAL VALUES AND RETURNS TO THE INVOKING
                   397 ;                      PROCEDURE.
                   398 ;
                   399 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   400 ;
 020E C1           401 RETPAR: POP     B               ;RESTORE B-C-REG FROM STACK
 020F D1           402         POP     D               ;RESTORE D-E-REG FROM STACK
 0210 E1           403         POP     H               ;RESTORE H-L-REG FROM STACK
 0211 F1           404         POP     PSW             ;RESTORE A-REG AND FLAGS FROM STACK
 0212 C9           405         RET                     ;RETURN TO INVOKING PROCEDURE
                   406         END                     ;END OF NOVAST MODULE
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVAST   PAGE    8

PUBLIC SYMBOLS
ESCAPE D 0066    NOVAST D 0000    RETPAR D 020E    TIMINT D 00A3

EXTERNAL SYMBOLS
BOTTOM E 0000    CHAAMT E 0000    CHAPTR E 0000    CURBLK E 0000    FSTBLK E 0000    LSTUSR E 0000
NOINST E 0000    NOVSTS E 0000    ROUPTR E 0000    RUNFLG E 0000    SNTP1  E 0000    STBY0  E 0000
SYSTAT E 0000    TIMCT1 E 0000    TIMCTR E 0000    USRFLG E 0000

USER SYMBOLS
BOTTOM E 0000    CHA1   D 01C8    CHAAMT E 0000    CHAOUT D 01A4    CHAPTR E 0000    CRCHA  D 01D5
CURBLK E 0000    CWE1   D 0120    CWECHO D 010E    ESCAPE D 0066    FSTBLK E 0000    LSTUSR E 0000
MT1    D 0153    MT2    D 0167    MT3    D 0170    MT4    D 0193    NOINST E 0000    NOVAST D 0000
NOVSTS E 0000    RETPAR D 020E    ROUPTR E 0000    RUNFLG E 0000    SNTP1  E 0000    STB1   D 002D
STB2   D 0035    STB3   D 0044    STB4   D 0057    STBCHK D 0029    STBY0  E 0000    SYSTAT E 0000
TEN1   D 01F0    TEN2   D 01F9    TEN3   D 0206    TENSEC D 01E3    TIM1   D 00BF    TIM2   D 00C8
TIM3   D 00D5    TIM4   D 00E5    TIM5   D 0100    TIMCT1 E 0000    TIMCTR E 0000    TIMINT D 00A3
TIMTST D 00B2    USRFLG E 0000

ASSEMBLY COMPLETE,    NO ERRORS
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                  PAGE    1

BOTTOM    16     248
CHA1     326     333#
CHAAMT    16     105     282
CHAOUT    42     312#
CHAPTR    16     101     219     298     316
CRCHA     44     350#
CURBLK    16     269
CWE1     226#    355
CWECHO    40     215#
ESCAPE    18      92#    195
FSTBLK    16     271
LSTUSR    16      60      79      88
MT1      249#    266
MT2      251     255     262#
MT3      261     267#
MT4      285     288     290#
NOINST    16      95     199     241
NOVAST    15      18      29#
NOVSTS    16      33     115     230     335     388
RETPAR    18      47      82     123     175     182     201     218     234     237     302     315     332     337
         353     387     390     401#
ROUPTR    16      97
RUNFLG    16     232     238
SNTP1     17      86     246
STB1      62#     81
STB2      64      66#
STB3      68      71      73#
STB4      77      83#
STBCHK    36      60#
STBY0     17      66
SYSTAT    17     183
TEN1     372     374#
TEN2     378     380#
TEN3     382     388#
TENSEC    46     367#
TIM1     160     162#
TIM2     166     168#
TIM3     170     176#
TIM4     179     183#
TIM5     191     196#
TIMCT1    17     111     135     156     368
TIMCTR    17     109     139     176
TIMINT    18     134#    181     229     334
TIMTST    38     155#
USRFLG    17      90     188     242

CROSS REFERENCE COMPLETE
```

```
ASM80 :F1:NOVA.SRC MACROFILE COND NOOBJECT XREF PAGELENGTH(70) PAGEWIDTH(107) PRINT(:LP:)


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    1


  LOC  OBJ         LINE        SOURCE STATEMENT

                      1 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                      2 ;
                      3 ;          MODULE NAME:  NOVA
                      4 ;
                      5 ;          DESCRIPTION:  THIS MODULE PROCESSES INTERRUPTS CAUSED BY
                      6 ;                        ECHOS RETURNING FROM THE NOVA.
                      7 ;
                      8 ;          MOST RECENT REVISION:  AUGUST 22, 1980
                      9 ;
                     10 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     11 ;
                     12         NAME    NOVA
                     13         DSEG
                     14         EXTRN   BOTTOM,CHAAMT,CHAPTR,CURBLK,EOIM,EOI534,FSTBLK
                     15         EXTRN   NOINST,NOVSTS,ROUPTR,RUNFLG,SENTO,SNTP1,STAT,STBY0,SYSTAT
                     16         EXTRN   TIMCT1,TIMCTR,TIMINT,TTYBUF,USRFLG
                     17         PUBLIC  NOVA
                     18 ;
                     19 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     20 ;
                     21 ;     ROUTINE NAME:  NOVA
                     22 ;     DESCRIPTION:  THIS ROUTINE CHECKS THE VALUE IN NOINST AND JUMPS
                     23 ;                   TO THE APPROPRIATE ONE OF SIX ROUTINES, DEPENDING
                     24 ;                   ON THE VALUE OF THAT FLAG.
                     25 ;
                     26 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     27 ;
  0000 F5            28 NOVA:   PUSH    PSW             ;STORE A-REG AND FLAGS IN STACK
  0001 E5            29         PUSH    H               ;STORE H-L-REG IN STACK
  0002 D5            30         PUSH    D               ;STORE D-E-REG IN STACK
  0003 C5            31         PUSH    B               ;STORE B-C-REG IN STACK
  0004 210000   E    32         LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
  0007 7E            33         MOV     A,M             ;MOVE NOINST INTO A-REG
  0008 FE00          34         CPI     00H             ;IS NOINST ZERO?
  000A CA2900   D    35         JZ      NOVTST          ;IF YES, JUMP TO NOVTST
  000D FE01          36         CPI     01H             ;IS NOINST 1?
  000F CA9001   D    37         JZ      NOVECH          ;IF YES, JUMP TO NOVECH
  0012 FE02          38         CPI     02H             ;IS NOINST 2?
  0014 CAFB01   D    39         JZ      NOVSGN          ;IF YES, JUMP TO NOVSGN
  0017 FE03          40         CPI     03H             ;IS NOINST 3?
  0019 CA7802   D    41         JZ      NOVXMT          ;IF YES, JUMP TO NOVXMT
  001C FE04          42         CPI     04H             ;IS NOINST 4?
  001E CA0703   D    43         JZ      NVECH2          ;IF YES, JUMP TO NVECH2
  0021 FE05          44         CPI     05H             ;IS NOINST 5?
  0023 CA2D04   D    45         JZ      NOVBYE          ;IF YES, JUMP TO NOVBYE
  0026 C3E804   D    46         JMP     RETSYS          ;IF NONE OF THE ABOVE, JUMP TO RETSYS
                     47 ;
                     48 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     49 ;
                     50 ;     ROUTINE NAME:  NOVTST
                     51 ;     DESCRIPTION:  THIS ROUTINE CHECKS WHAT COMES BACK FROM THE NOVA
                     52 ;                   AFTER AN ESCAPE HAS BEEN SENT.
                     53 ;
                     54 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     55 ;
  0029 210000   E    56 NOVTST: LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  002C 7E            57         MOV     A,M             ;MOVE CHAPTR-MSB INTO A REG
  002D 57            58         MOV     D,A             ;MOVE CHAPTR-MSB INTO D-REG
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    2

 LOC  OBJ         LINE        SOURCE STATEMENT

 002E FE00          59         CPI    00H             ;IS CHAPTR-MSB ZERO?
 0030 C2AC00   D    60         JNZ    TST4            ;IF NO, JUMP TO TST4
 0033 23            61         INX    H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 0034 7E            62         MOV    A,M             ;MOVE CHAPTR-LSB INTO A-REG
 0035 5F            63         MOV    E,A             ;MOVE CHAPTR-LSB INTO E-REG
 0036 FE00          64         CPI    00H             ;IS CHAPTR-LSB ZERO?
 0038 C2AC00   D    65         JNZ    TST4            ;IF NO, JUMP TO TST4
 003B DB16          66         IN     16H             ;GET CHARACTER FROM NOVA THROUGH USART-3
 003D E67F          67         ANI    7FH             ;RESET PARITY BIT
 003F FE42          68         CPI    42H             ;IS CHARACTER A 'B'?
 0041 C25900   D    69         JNZ    TST1            ;IF NO, JUMP TO TST1
 0044 210000   E    70         LXI    H,ROUPTR        ;LOAD ADDRESS OF ROUPTR INTO H-L-REG
 0047 111D01   D    71         LXI    D,BUSY          ;LOAD ADDRESS OF BUSY INTO D-E-REG
 004A CDE500   D    72         CALL   TSTSB1          ;MOVE ADDRESS OF BUSY INTO ROUPTR IN MEMORY
 004D 210000   E    73         LXI    H,CHAPTR        ;LOAD ADDRESS OF CHAPTR INTO H-L-REG
 0050 11ED04   D    74         LXI    D,USY           ;LOAD ADDRESS OF USY INTO D-E-REG
 0053 CDE500   D    75         CALL   TSTSB1          ;MOVE ADDRESS OF USY INTO CHAPTR IN MEMORY
 0056 C3E804   D    76         JMP    RETSYS          ;JUMP TO RETSYS
 0059 FE2A          77 TST1:   CPI    2AH             ;IS CHARACTER AN ASTERISK?
 005B CAF800   D    78         JZ     ONLINE          ;IF YES, JUMP TO ONLINE
 005E FE52          79         CPI    52H             ;IS CHARACTER AN 'R'?
 0060 C27800   D    80         JNZ    TST2            ;IF NO, JUMP TO TST2
 0063 210000   E    81         LXI    H,ROUPTR        ;LOAD ADDRESS OF ROUPTR INTO H-L-REG
 0066 112E01   D    82         LXI    D,READY         ;LOAD ADDRESS OF READY INTO D-E-REG
 0069 CDE500   D    83         CALL   TSTSB1          ;MOVE ADDRESS OF READY INTO ROUPTR IN MEMORY
 006C 210000   E    84         LXI    H,CHAPTR        ;MOVE ADDRESS OF CHAPTR INTO H-L-REG
 006F 11F104   D    85         LXI    D,EADY          ;LOAD ADDRESS OF EADY INTO D-E-REG
 0072 CDE500   D    86         CALL   TSTSB1          ;MOVE ADDRESS OF EADY INTO CHAPTR IN MEMORY
 0075 C3E804   D    87         JMP    RETSYS          ;JUMP TO RETSYS
 0078 FE41          88 TST2:   CPI    41H             ;IS CHARACTER AN 'A'?
 007A C29200   D    89         JNZ    TST3            ;IF NO, JUMP TO TST3
 007D 210000   E    90         LXI    H,ROUPTR        ;LOAD ADDRESS OF ROUPTR INTO H-L-REG
 0080 116401   D    91         LXI    D,ACCID         ;LOAD ADDRESS OF ACCID INTO D-E-REG
 0083 CDE500   D    92         CALL   TSTSB1          ;MOVE ADDRESS OF ACCID INTO ROUPTR IN MEMORY
 0086 210000   E    93         LXI    H,CHAPTR        ;LOAD ADDRESS OF CHAPTR INTO H-L-REG
 0089 11F604   D    94         LXI    D,CCOU          ;LOAD ADDRESS OF CCOU INTO D-E-REG
 008C CDE500   D    95         CALL   TSTSB1          ;MOVE ADDRESS OF CCOU INTO CHAPTR
 008F C3E804   D    96         JMP    RETSYS          ;JUMP TO RETSYS
 0092 FE4D          97 TST3:   CPI    4DH             ;IS CHARACTER AN 'M'?
 0094 C2E804   D    98         JNZ    RETSYS          ;IF NO, JUMP TO RETSYS
 0097 210000   E    99         LXI    H,ROUPTR        ;LOAD ADDRESS OF ROUPTR INTO H-L-REG
 009A 111D01   D   100         LXI    D,BUSY          ;LOAD ADDRESS OF BUSY INTO D-E-REG
 009D CDE500   D   101         CALL   TSTSB1          ;MOVE ADDRESS OF BUSY INTO ROUPTR IN MEMORY
 00A0 210000   E   102         LXI    H,CHAPTR        ;LOAD ADDRESS OF CHAPTR INTO H-L-REG
 00A3 11FB04   D   103         LXI    D,AX            ;LOAD ADDRESS OF AX INTO D-E-REG
 00A6 CDE500   D   104         CALL   TSTSB1          ;MOVE ADDRESS OF AX INTO CHAPTR IN MEMORY
 00A9 C3E804   D   105         JMP    RETSYS          ;JUMP TO RETSYS
 00AC DB16         106 TST4:   IN     16H             ;GET CHARACTER FROM NOVA THROUGH USART-3
 00AE E67F         107         ANI    7FH             ;RESET PARITY BIT
 00B0 210000   E   108         LXI    H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 00B3 56           109         MOV    D,M             ;MOVE CHAPTR-MSB INTO D-REG
 00B4 23           110         INX    H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 00B5 5E           111         MOV    E,M             ;MOVE CHAPTR-LSB INTO E-REG
 00B6 EB           112         XCHG                   ;SWAP CHAPTR INTO H-L-REG
 00B7 BE           113         CMP    M               ;IS CHARACTER AT CHAPTR EQUAL TO CHARACTER FROM NOV
                       A?
 00B8 CAC100   D   114         JZ     TST5            ;IS YES, JUMP TO TST5
 00BB CDE900   D   115         CALL   TSTSB2          ;SET CHAPTR AND ROUPTR TO ZERO
 00BE C3E804   D   116         JMP    RETSYS          ;JUMP TO RETSYS
```

EPA Form 2350-4 (4-80)
(PREVIOUSLY CIN EPA FORM 547)
TYPING GUIDE SHEET

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    3

  LOC  OBJ         LINE        SOURCE STATEMENT

  00C1 210000   E    117 TST5:   LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  00C4 56            118         MOV     D,M             ;MOVE CHAPTR-MSB INTO D-REG
  00C5 23            119         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  00C6 5E            120         MOV     E,M             ;MOVE CHAPTR-LSB INTO E-REG
  00C7 13            121         INX     D               ;INCREMENT CHAPTR IN D-E-REG
  00C8 73            122         MOV     M,E             ;UPDATE CHAPTR-LSB IN MEMORY
  00C9 2B            123         DCX     H               ;DECREMENT H-L-REG TO ADDRESS OF CHAPTR-MSB
  00CA 72            124         MOV     M,D             ;UPDATE CHAPTR-MSB IN MEMORY
  00CB EB            125         XCHG                    ;MOVE CHAPTR TO H-L-REG
  00CC 7E            126         MOV     A,M             ;MOVE CHARACTER AT CHAPTR INTO A-REG
  00CD E67F          127         ANI     7FH             ;RESET PARITY BIT
  00CF FE0D          128         CPI     0DH             ;IS CHARACTER A CARRIAGE RETURN?
  00D1 C2E804   D    129         JNZ     RETSYS          ;IF NO, JUMP TO RETSYS
  00D4 210000   E    130         LXI     H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
  00D7 56            131         MOV     D,M             ;MOVE ROUPTR-MSB INTO D-REG
  00D8 23            132         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
  00D9 5E            133         MOV     E,M             ;MOVE ROUPTR-LSB INTO E-REG
  00DA CDE900   D    134         CALL    TSTSB2          ;SET CHAPTR AND ROUPTR TO ZERO
  00DD DB19          135         IN      19H             ;READ CURRENT INTERRUPT MASK REG FROM PIC-0
  00DF F680          136         ORI     80H             ;SET MASK BIT FOR NOVA INTERRUPT (INTERRUPT-6)
  00E1 D319          137         OUT     19H             ;PROGRAM PIC-0 TO MASK NOVA INTERRUPT
  00E3 EB            138         XCHG                    ;MOVE ROUPTR TO H-L-REG
  00E4 E9            139         PCHL                    ;MOVE ROUPTR TO PROGRAM COUNTER
  00E5 72            140 TSTSB1: MOV     M,D             ;MOVE MSB TO MEMORY
  00E6 23            141         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF LSB
  00E7 73            142         MOV     M,E             ;MOVE LSB TO MEMORY
  00E8 C9            143         RET                     ;RETURN TO INVOKING PROCEDURE
  00E9 3E00          144 TSTSB2: MVI     A,00H           ;SET A-REG TO ZERO
  00EB 210000   E    145         LXI     H,CHAPTR        ;LOAD H-L-REG WITH ADDRESS OF CHAPTR-MSB
  00EE 77            146         MOV     M,A             ;SET CHAPTR-MSB TO ZERO
  00EF 23            147         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF LSB OF CHAPTR
  00F0 77            148         MOV     M,A             ;SET CHAPTR-LSB TO ZERO
  00F1 210000   E    149         LXI     H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
  00F4 77            150         MOV     M,A             ;SET ROUPTR-MSB TO ZERO
  00F5 23            151         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
  00F6 77            152         MOV     M,A             ;SET ROUPTR-LSB TO ZERO
  00F7 C9            153         RET                     ;RETURN TO INVOKING PROCEDURE
                     154 ;
                     155 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     156 ;
                     157 ;      ROUTINE NAME:  ONLINE
                     158 ;      DESCRIPTION:   THIS ROUTINE IS INVOKED IF THE SYSTEM SENDS AN
                     159 ;                     ESCAPE TO THE NOVA AND FINDS THAT THE PORT IT IS
                     160 ;                     USING IS ALREADY LOGGED IN.  THE ROUTINE LOGS OFF
                     161 ;                     FROM THE NOVA.
                     162 ;
                     163 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     164 ;
  00F8 012F01   D    165 ONLINE: LXI     B,NOVESC        ;LOAD ADDRESS OF NOVESC INTO B-D-REG
  00FB 210000   E    166         LXI     H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
  00FE 70            167         MOV     M,B             ;MOVE MSB OF ADDRESS OF NOVESC INTO ROUPTR-MSB
  00FF 23            168         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
  0100 71            169         MOV     M,C             ;MOVE LSB OF ADDRESS OF NOVESC INTO ROUPTR-LSB
  0101 01FE04   D    170         LXI     B,BYE           ;LOAD ADDRESS OF BYE INTO B-C-REG
  0104 210000   E    171         LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-LSB INTO H-L-REG
  0107 70            172         MOV     M,B             ;MOVE MSB OF ADDRESS OF BYE INTO CHAPTR-MSB
  0108 23            173         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  0109 71            174         MOV     M,C             ;MOVE LSB OF ADDRESS OF BYE INTO CHAPTR-LSB
  010A 210000   E    175         LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    4


  LOC  OBJ          LINE        SOURCE STATEMENT

  010D 3601          176         MVI     M,01H          ;SET NOINST TO 1
  010F 210000   E    177         LXI     H,NOVSTS       ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
  0112 3602          178         MVI     M,02H          ;SET NOVSTS TO 2
  0114 DB19          179         IN      19H            ;READ CURRENT INTERRUPT MASK REG FROM PIC-0
  0116 E6BF          180         ANI     0BFH           ;RESET MASK BIT FOR NOVA INTERRUPT (INTERRUPT-6)
  0118 D319          181         OUT     19H            ;PROGRAM PIC-0 TO UNMASK NOVA INTERRUPT
  011A C3E804   D    182         JMP     RETSYS         ;JUMP TO RETSYS
                     183 ;
                     184 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     185 ;
                     186 ;      ROUTINE NAME:  BUSY
                     187 ;      DESCRIPTION:  THIS ROUTINE IS INVOKED IF THE SYSTEM SENDS AN
                     188 ;                    ESCAPE TO THE NOVA AND GETS AN 'ALL LINES BUSY'
                     189 ;                    OR 'MAX USERS' MESSAGE BACK.  THE ROUTINE TURNS ON
                     190 ;                    THE APPROPRIATE ERROR BIT AND SETS THE TIMER TO
                     191 ;                    SEND ANOTHER ESCAPE IN TWO MINUTES.
                     192 ;
                     193 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     194 ;
  011D 3E02          195 BUSY:   MVI     A,02H          ;SET TIME AT 2 MINUTES
  011F CD0000   E    196         CALL    TIMINT         ;INITIALIZE TIMER
  0122 210000   E    197         LXI     H,SYSTAT       ;LOAD ADDRESS OF SYSTAT INTO H-L-REG
  0125 7E            198         MOV     A,M            ;MOVE SYSTAT INTO A-REG
  0126 E601          199         ANI     01H            ;RESET ALL ERROR BITS EXCEPT MEMORY OVERFLOW
  0128 F604          200         ORI     04H            ;SET BIT FOR ALL LINES BUSY OR MAX USERS
  012A 77            201         MOV     M,A            ;UPDATE SYSTAT IN MEMORY
  012B C3E804   D    202         JMP     RETSYS         ;JUMP TO RETSYS
                     203 ;
                     204 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     205 ;
                     206 ;      ROUTINE NAME:  READY
                     207 ;      DESCRIPTION:  THIS ROUTINE IS INVOKED IF THE SYSTEM SENDS AN
                     208 ;                    ESCAPE TO THE NOVA AND GETS A 'LATS READY' MESSAGE
                     209 ;                    IN RETURN.  THE ROUTINE SENDS ANOTHER ESCAPE TO
                     210 ;                    THE NOVA.
                     211 ;
                     212 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     213 ;
  012E 00            214 READY:  NOP                    ;LABEL ONLY
  012F 3E27          215 NOVESC: MVI     A,27H          ;MOVE USART COMMAND WORD INTO A-REG
  0131 D317          216         OUT     17H            ;PROGRAM USART-3 TO SET DATA TERMINAL READY BIT
  0133 3E00          217         MVI     A,00H          ;MOVE ZERO INTO A-REG
  0135 210000   E    218         LXI     H,NOINST       ;LOAD ADDRESS OF NOINST INTO H-L-REG
  0138 77            219         MOV     M,A            ;SET NOINST TO ZERO
  0139 210000   E    220         LXI     H,ROUPTR       ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
  013C 77            221         MOV     M,A            ;SET ROUPTR-MSB TO ZERO
  013D 23            222         INX     H              ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
  013E 77            223         MOV     M,A            ;SET ROUPTR-LSB TO ZERO
  013F 210000   E    224         LXI     H,CHAPTR       ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  0142 77            225         MOV     M,A            ;SET CHAPTR-MSB TO ZERO
  0143 23            226         INX     H              ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  0144 77            227         MOV     M,A            ;SET CHAPTR-LSB TO ZERO
  0145 210000   E    228         LXI     H,CHAAMT       ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
  0148 77            229         MOV     M,A            ;SET CHAAMT-MSB TO ZERO
  0149 23            230         INX     H              ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
  014A 77            231         MOV     M,A            ;SET CHAAMT-LSB TO ZERO
  014B 3E01          232         MVI     A,01H          ;SET TIME AT 1 MINUTES
  014D CD0000   E    233         CALL    TIMINT         ;INITIALIZE TIMER
  0150 210000   E    234         LXI     H,NOVSTS       ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    5

  LOC  OBJ         LINE        SOURCE STATEMENT

  0153 3601         235          MVI     M,01H           ;SET NOVSTS TO 1
  0155 DB19         236          IN      19H             ;READ CURRENT INTERRUPT MASK REGISTER FROM PIC-0
  0157 E6BF         237          ANI     0BFH            ;RESET MASK BIT FOR NOVA INTERRUPT (INTERRUPT-6)
  0159 D319         238          OUT     19H             ;PROGRAM PIC-0 TO UNMASK NOVA INTERRUPT
  015B 3E1B         239          MVI     A,1BH           ;MOVE ESC CHARACTER INTO A-REG
  015D F680         240          ORI     80H             ;SET PARITY BIT
  015F D316         241          OUT     16H             ;SEND ESCAPE TO NOVA THROUGH USART-3
  0161 C3E804   D   242          JMP     RETSYS          ;JUMP TO RETSYS
                    243 ;
                    244 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    245 ;
                    246 ;       ROUTINE NAME:  ACCID
                    247 ;       DESCRIPTION:   THIS ROUTINE IS INVOKED WHEN THE SYSTEM SENDS AN
                    248 ;                      ESCAPE TO THE NOVA AND GETS AN 'ACCOUNT ID' MESSAGE
                    249 ;                      IN RETURN.  THE ROUTINE GETS THE APPROPRIATE ACCOUNT
                    250 ;                      ID SET UP FOR TRANSMISSION TO THE NOVA.
                    251 ;
                    252 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    253 ;
  0164 210000   E   254 ACCID:   LXI     H,USRFLG        ;LOAD ADDRESS OF USRFLG INTO H-L-REG
  0167 7E           255          MOV     A,M             ;MOVE USRFLG TO A-REG
  0168 010A00   E   256          LXI     B,TTYBUF+0AH    ;LOAD ADDRESS OF ID0 INTO B-C-REG
  016B FE00         257          CPI     00H             ;IS USRFLG ZERO?
  016D CA7701   D   258          JZ      ACC1            ;IF YES, JUMP TO ACC1
  0170 04           259          INR     B               ;INCREMENT B-REG TO PRODUCE ADDRESS OF ID1
  0171 FE01         260          CPI     01H             ;IS USRFLG 1?
  0173 CA7701   D   261          JZ      ACC1            ;IF YES, THEM JUMP TO ACC1
  0176 04           262          INR     B               ;INCREMENT B-REG TO PRODUCE ADDRESS OF ID2
  0177 210000   E   263 ACC1:    LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  017A 70           264          MOV     M,B             ;MOVE MSB OF ADDRESS OF IDn INTO CHAPTR-MSB
  017B 23           265          INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  017C 71           266          MOV     M,C             ;MOVE LSB OF ADDRESS OF IDn INTO CHAPTR-LSB
  017D 210000   E   267          LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
  0180 3603         268          MVI     M,03H           ;SET NOVSTS TO 3
  0182 210000   E   269          LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
  0185 3602         270          MVI     M,02H           ;SET NOINST TO 2
  0187 DB19         271          IN      19H             ;READ CURRENT INTERRUPT MASK REG FROM PIC-0
  0189 E6BF         272          ANI     0BFH            ;RESET MASK BIT FOR NOVA INTERRUPT (INTERRUPT-6)
  018B D319         273          OUT     19H             ;PROGRAM PIC-0 TO UNMASK NOVA INTERRUPT
  018D C3E804   D   274          JMP     RETSYS          ;JUMP TO RETSYS
                    275 ;
                    276 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    277 ;
                    278 ;       ROUTINE NAME:  NOVECH
                    279 ;       DESCRIPTION:   THIS IS THE MAIN ROUTINE FOR PROCESSING CHARACTERS
                    280 ;                      ECHOED BY THE NOVA.
                    281 ;
                    282 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    283 ;
  0190 DB16         284 NOVECH:  IN      16H             ;GET CHARACTER FROM NOVA THROUGH USART-3
  0192 E67F         285          ANI     7FH             ;RESET PARITY BIT
  0194 210000   E   286          LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  0197 56           287          MOV     D,M             ;MOVE CHAPTR-MSB INTO D-REG
  0198 23           288          INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  0199 5E           289          MOV     E,M             ;MOVE CHAPTR-LSB INTO E-REG
  019A EB           290          XCHG                    ;SWAP CHAPTR INTO H-L-REG
  019B BE           291          CMP     M               ;IS CHARACTER AT CHAPTR EQUAL TO CHARACTER FROM NOV
A?
  019C CAB701   D   292          JZ      ECH1            ;IF YES, JUMP TO ECH1
```

BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
CHOPPED HEAD BEGIN SECTIONS HERE

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    6

 LOC  OBJ         LINE        SOURCE STATEMENT

 019F E67F         293           ANI    7FH          ;RESET PARITY BIT
 01A1 FE5F         294           CPI    5FH          ;IS CHARACTER FROM NOVA AN UNDERSCORE?
 01A3 C2E804   D   295           JNZ    RETSYS       ;IF NO, JUMP TO RETSYS
 01A6 210000   E   296           LXI    H,CHAAMT     ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
 01A9 3E00         297           MVI    A,00H        ;SET A-REG EQUAL TO ZERO
 01AB BE           298           CMP    M            ;IS CHAAMT-MSB ZERO?
 01AC C24603   D   299           JNZ    BBBBB        ;IF NO, JUMP TO BBBBB
 01AF 23           300           INX    H            ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
 01B0 BE           301           CMP    M            ;IS CHAAMT-LSB ZERO?
 01B1 C24603   D   302           JNZ    BBBBB        ;IF NO, JUMP TO BBBBB
 01B4 C3E804   D   303           JMP    RETSYS       ;IF YES, JUMP TO RETSYS
 01B7 E67F         304 ECH1:     ANI    7FH          ;RESET PARITY BIT
 01B9 FE0D         305           CPI    0DH          ;IS CHARACTER A CARRIAGE RETURN?
 01BB C2D001   D   306           JNZ    ECH2         ;IF NO, JUMP TO ECH2
 01BE 3E14         307           MVI    A,14H        ;SET TIME FOR 20 MINUTES
 01C0 CD0000   E   308           CALL   TIMINT       ;INITIALIZE TIMER
 01C3 210000   E   309           LXI    H,NOVSTS     ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 01C6 3601         310           MVI    M,01H        ;SET NOVSTS TO 1
 01C8 210000   E   311           LXI    H,ROUPTR     ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
 01CB 56           312           MOV    D,M          ;MOVE ROUPTR-MSB INTO D-REG
 01CC 23           313           INX    H            ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
 01CD 5E           314           MOV    E,M          ;MOVE ROUPTR-LSB INTO E-REG
 01CE EB           315           XCHG                ;SWAP ROUPTR INTO H-L-REG
 01CF E9           316           PCHL                ;LOAD ROUPTR INTO PROGRAM COUNTER
 01D0 FE3E         317 ECH2:     CPI    3EH          ;IS CHARACTER A RIGHT ANGLE BRACKET?
 01D2 C2E901   D   318           JNZ    ECH3         ;IF NO, JUMP TO ECH3
 01D5 210000   E   319           LXI    H,NOINST     ;LOAD ADDRESS OF NOINST INTO H-L-REG
 01D8 3604         320           MVI    M,04H        ;SET NOINST TO 4
 01DA 210000   E   321           LXI    H,ROUPTR     ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
 01DD 56           322           MOV    D,M          ;MOVE ROUPTR-MSB INTO D-REG
 01DE 23           323           INX    H            ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
 01DF 5E           324           MOV    E,M          ;MOVE ROUPTR-LSB INTO E-REG
 01E0 210000   E   325           LXI    H,CHAPTR     ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 01E3 72           326           MOV    M,D          ;UPDATE CHAPTR-MSB
 01E4 23           327           INX    H            ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 01E5 73           328           MOV    M,E          ;UPDATE CHAPTR-LSB
 01E6 C32403   D   329           JMP    AAAAA        ;JUMP TO AAAAA
 01E9 210000   E   330 ECH3:     LXI    H,CHAPTR     ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 01EC 56           331           MOV    D,M          ;MOVE CHAPTR-MSB INTO D-REG
 01ED 23           332           INX    H            ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 01EE 5E           333           MOV    E,M          ;MOVE CHAPTR-LSB INTO E-REG
 01EF 13           334           INX    D            ;INCREMENT CHAPTR IN D-E-REG
 01F0 73           335           MOV    M,E          ;MOVE CHAPTR-LSB BACK TO MEMORY
 01F1 2B           336           DCX    H            ;DECREMENT H-L-REG BACK TO ADDRESS OF CHAPTR-MSB
 01F2 72           337           MOV    M,D          ;MOVE CHAPTR-MSB BACK TO MEMORY
 01F3 210000   E   338           LXI    H,NOVSTS     ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 01F6 3602         339           MVI    M,02H        ;SET NOVSTS TO 2
 01F8 C3E804   D   340           JMP    RETSYS       ;JUMP TO RETSYS
                   341 ;
                   342 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   343 ;
                   344 ;     ROUTINE NAME:  NOVSGN
                   345 ;     DESCRIPTION:   THIS ROUTINE IS INVOKED WHEN THE NOVA RESPONDS TO
                   346 ;                    THE SYSTEM SENDING THE ACCOUNT ID.  IF THE NOVA
                   347 ;                    RESPONDS WITH 'UNKNOWN ID' OR 'ID IN USE', THE
                   348 ;                    SYSTEM SETS THE APPROPRIATE ERROR BIT AND SETS THE
                   349 ;                    TIMER TO TRY AGAIN IN TWO MINUTES.  OTHERWISE, THE
                   350 ;                    SYSTEM SETS UP THE MESSAGE 'RUN"MICROINPUT' FOR
                   351 ;                    TRANSMISSION TO THE NOVA.  IT ALSO UPDATES THE
```

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
PAGE NUMBER
EPA Form 2350-6 (4-80)
(PREVIOUSLY EPA FORM 381)
TYPING GUIDE SHEET

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    7

  LOC  OBJ         LINE        SOURCE STATEMENT

                    352 ;                   USER'S STATUS BYTE TO INDICATE THAT THE SYSTEM
                    353 ;                   IS TRANSMITTING FOR THAT USER.
                    354 ;
                    355 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    356 ;
 01FB DB16          357 NOVSGN: IN      16H             ;GET CHARACTER FROM NOVA THROUGH USART-3
 01FD E67F          358         ANI     7FH             ;RESET PARITY BIT
 01FF FE2A          359         CPI     2AH             ;IS CHARACTER AN ASTERISK?
 0201 CA2002    D   360         JZ      SGN1            ;IF YES, JUMP TO SGN1
 0204 FE55          361         CPI     55H             ;IS CHARACTER A 'U'?
 0206 C2E804    D   362         JNZ     RETSYS          ;IF NO, JUMP TO RETSYS
 0209 DB19          363         IN      19H             ;READ CURRENT INTERRUPT MASK REGISTER FROM PIC-0
 020B F680          364         ORI     80H             ;SET NOVA INTERRUPT MASK BIT (INTERRUPT-6)
 020D D319          365         OUT     19H             ;PROGRAM PIC-0 TO MASK NOVA INTERRUPT
 020F 210000    E   366         LXI     H,SYSTAT        ;LOAD ADDRESS OF SYSTAT INTO H-L-REG
 0212 7E            367         MOV     A,M             ;MOVE SYSTAT INTO A-REG
 0213 E601          368         ANI     01H             ;RESET ALL ERROR BITS EXCEPT MEMORY OVERFLOW
 0215 F602          369         ORI     02H             ;SET BIT FOR 'ID IN USE' OR 'UNKNOWN ID' ERROR
 0217 77            370         MOV     M,A             ;UPDATE SYSTAT
 0218 3E02          371         MVI     A,02H           ;SET TIME FOR TWO MINUTES
 021A CD0000    E   372         CALL    TIMINT          ;INITIALIZE TIMER
 021D C3E804    D   373         JMP     RETSYS          ;JUMP TO RETSYS
 0220 210000    E   374 SGN1:   LXI     H,SYSTAT        ;LOAD ADDRESS OF SYSTAT INTO H-L-REG
 0223 7E            375         MOV     A,M             ;MOVE SYSTAT INTO A-REG
 0224 E601          376         ANI     01H             ;RESET ALL ERROR BITS EXCEPT MEMORY OVERFLOW
 0226 77            377         MOV     M,A             ;UPDATE SYSTAT
 0227 110205    D   378         LXI     D,RUN           ;LOAD ADDRESS OF RUN INTO D-E-REG
 022A 210000    E   379         LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 022D 72            380         MOV     M,D             ;MOVE MSB OF ADDRESS OF RUN TO CHAPTR-MSB
 022E 23            381         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 022F 73            382         MOV     M,E             ;MOVE LSB OF ADDRESS OF RUN TO CHAPTR-LSB
 0230 210000    E   383         LXI     H,RUNFLG        ;LOAD ADDRESS OF RUNFLG INTO H-L-REG
 0233 36FF          384         MVI     M,0FFH          ;SET RUNFLG TO FF
 0235 210000    E   385         LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
 0238 3601          386         MVI     M,01H           ;SET NOINST TO 1
 023A 210000    E   387         LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 023D 3602          388         MVI     M,02H           ;SET NOVSTS TO 2
 023F 210000    E   389         LXI     H,USRFLG        ;LOAD ADDRESS OF USRFLG INTO H-L-REG
 0242 7E            390         MOV     A,M             ;MOVE USRFLG INTO A-REG
 0243 210000    E   391         LXI     H,STAT          ;LOAD ADDRESS OF STAT0 INTO H-L-REG
 0246 FE00          392         CPI     00H             ;IS USRFLG ZERO?
 0248 CA5202    D   393         JZ      SGN2            ;IF YES, JUMP TO SGN2
 024B 24            394         INR     H               ;INCREMENT H-REG TO FORM ADDRESS OF STAT1
 024C FE01          395         CPI     01H             ;IS USRFLG 1?
 024E CA5202    D   396         JZ      SGN2            ;IF YES, JUMP TO SGN2
 0251 24            397         INR     H               ;INCREMENT H-REG TO FORM ADDRESS OF STAT2
 0252 F5            398 SGN2:   PUSH    PSW             ;STORE USRFLG IN STACK
 0253 7E            399         MOV     A,M             ;MOVE STATn INTO A-REG
 0254 F608          400         ORI     08H             ;SET uTRANS BIT
 0256 77            401         MOV     M,A             ;UPDATE STATn
 0257 F1            402         POP     PSW             ;RESTORE USRFLG TO A-REG
 0258 E5            403         PUSH    H               ;STORE ADDRESS OF STATn
 0259 210000    E   404         LXI     H,STBY0         ;LOAD ADDRESS OF STBY0 INTO H-L-REG
 025C FE00          405         CPI     00H             ;IS USRFLG ZERO?
 025E CA6802    D   406         JZ      SGN3            ;IF YES, JUMP TO SGN3
 0261 24            407         INR     H               ;INCREMENT H-REG TO FORM ADDRESS OF STBY1
 0262 FE01          408         CPI     01H             ;IS USRFLG 1?
 0264 CA6802    D   409         JZ      SGN3            ;IF YES, JUMP TO SGN3
 0267 24            410         INR     H               ;INCREMENT H-REG TO FORM ADDRESS OF STBY2
```

```
  LOC OBJ         LINE        SOURCE STATEMENT

  0268 7E          411 SGN3:   MOV     A,M             ;MOVE STBYn INTO A-REG
  0269 210000   E  412         LXI     H,SNTP1         ;LOAD ADDRESS OF SNTP1
  026C BE          413         CMP     M               ;IS STBYn EQUAL TO SNTP1?
  026D E1          414         POP     H               ;RESTORE ADDRESS OF STATn TO H-L-REG
  026E C2E804   D  415         JNZ     RETSYS          ;IF STBYn IS NOT EQUAL TO SNTP1, JUMP TO RETSYS
  0271 7E          416         MOV     A,M             ;MOVE STATn INTO A-REG
  0272 E6FB        417         ANI     0FBH            ;RESET uSTBY BIT
  0274 77          418         MOV     M,A             ;UPDATE STATn
  0275 C3E804   D  419         JMP     RETSYS          ;JUMP TO RETSYS
                   420 ;
                   421 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   422 ;
                   423 ;      ROUTINE NAME:  NOVXMT
                   424 ;      DESCRIPTION:   THIS ROUTINE IS INVOKED WHEN A QUESTION MARK
                   425 ;                     IS RECEIVED FROM THE NOVA, PROMPTING FOR A NEW
                   426 ;                     LINE OF DATA.  THIS ROUTINE SETS UP THE FIRST CHARACTER
                   427 ;                     IN THE LINE FOR TRANSMISSION.  IT SKIPS OVER ANY BAD
                   428 ;                     CHARACTERS AT THE BEGINNING OF THE LINE, AND CHANGES
                   429 ;                     A LEADING BLANK (IF ANY) INTO A <32> WHICH
                   430 ;                     THE NOVA CAN INTERPRET CORRECTLY.
                   431 ;
                   432 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                   433 ;
  0278 DB16        434 NOVXMT: IN      16H             ;GET CHARACTER FROM NOVA THROUGH USART-3
  027A E67F        435         ANI     7FH             ;RESET PARITY BIT
  027C FE3F        436         CPI     3FH             ;IS CHARACTER A QUESTION MARK?
  027E C2E804   D  437         JNZ     RETSYS          ;IF NO, JUMP TO RETSYS
  0281 3E00        438         MVI     A,00H           ;MOVE ZERO INTO A-REG
  0283 210000   E  439         LXI     H,CHAAMT        ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
  0286 BE          440         CMP     M               ;IS CHAAMT-MSB A ZERO?
  0287 C28F02   D  441         JNZ     XMT1            ;IF NO, JUMP TO XMT1A
  028A 23          442         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
  028B BE          443         CMP     M               ;IS CHAAMT-LSB A ZERO?
  028C CAD503   D  444         JZ      CCCCC           ;IF YES, JUMP TO CCCCC
  028F 210000   E  445 XMT1:   LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  0292 56          446         MOV     D,M             ;MOVE CHAPTR-MSB INTO D-REG
  0293 23          447         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  0294 5E          448         MOV     E,M             ;MOVE CHAPTR-LSB INTO E-REG
  0295 EB          449         XCHG                    ;SWAP CHAPTR INTO H-L-REG
  0296 7E          450         MOV     A,M             ;MOVE CHARACTER AT CHAPTR INTO A-REG
  0297 E67F        451         ANI     7FH             ;RESET PARITY BIT
  0299 FE00        452         CPI     00H             ;IS CHARACTER A NULL?
  029B CACE02   D  453         JZ      XMT2            ;IF YES, JUMP TO XMT2
  029E FE05        454         CPI     05H             ;IS CHARACTER A CONTROL-E?
  02A0 CACE02   D  455         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02A3 FE0A        456         CPI     0AH             ;IS CHARACTER A LINEFEED?
  02A5 CACE02   D  457         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02A8 FE13        458         CPI     13H             ;IS CHARACTER AN X-OFF?
  02AA CACE02   D  459         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02AD FE1B        460         CPI     1BH             ;IS CHARACTER AN ESCAPE?
  02AF CACE02   D  461         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02B2 FE22        462         CPI     22H             ;IS CHARACTER A QUOTATION MARK?
  02B4 CACE02   D  463         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02B7 FE2C        464         CPI     2CH             ;IS CHARACTER A COMMA?
  02B9 CACE02   D  465         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02BC FE5C        466         CPI     5CH             ;IS CHARACTER A BACKSLASH?
  02BE CACE02   D  467         JZ      XMT2            ;IF YES, JUMP TO XMT2
  02C1 FE7F        468         CPI     7FH             ;IS CHARACTER A RUBOUT?
  02C3 CACE02   D  469         JZ      XMT2            ;IF YES, JUMP TO XMT2
```

EPA Form 2250-4 (4-69)

TYPING GUIDE SHEET

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE    9

 LOC  OBJ          LINE        SOURCE STATEMENT

 02C6 FE20          470         CPI     20H              ;IS CHARACTER A SPACE?
 02C8 CAEA02   D    471         JZ      XMT3             ;IF YES, JUMP TO XMT3
 02CB C34603   D    472         JMP     BBBBB            ;IF NONE OF THE ABOVE, JUMP TO BBBBB
 02CE 23            473 XMT2:   INX     H                ;INCREMENT CHAPTR
 02CF EB            474         XCHG                     ;SWAP CHAPTR BACK TO D-E-REG
 02D0 73            475         MOV     M,E              ;UPDATE CHAPTR-LSB
 02D1 2B            476         DCX     H                ;DECREMENT H-L-REG TO ADDRESS OF CHAPTR-MSB
 02D2 72            477         MOV     M,D              ;UPDATE CHAPTR-MSB
 02D3 210000   E    478         LXI     H,CHAAMT         ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
 02D6 56            479         MOV     D,M              ;MOVE CHAAMT-MSB INTO D-REG
 02D7 23            480         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
 02D8 5E            481         MOV     E,M              ;MOVE CHAAMT-LSB INTO E-REG
 02D9 1B            482         DCX     D                ;DECREMENT CHAAMT IN D-E-REG
 02DA 73            483         MOV     M,E              ;MOVE CHAAMT-LSB BACK TO MEMORY
 02DB 2B            484         DCX     H                ;DECREMENT H-L-REG BACK TO ADDRESS OF CHAAMT-MSB
 02DC 72            485         MOV     M,D              ;MOVE CHAAMT-MSB BACK TO MEMORY
 02DD 3E00          486         MVI     A,00H            ;SET A-REG TO ZERO
 02DF BA            487         CMP     D                ;IS CHAAMT-MSB ZERO?
 02E0 C28F02   D    488         JNZ     XMT1             ;IF NO, JUMP BACK TO XMT1
 02E3 BB            489         CMP     E                ;IS CHAAMT-LSB ZERO?
 02E4 C28F02   D    490         JNZ     XMT1             ;IF NO, JUMP BACK TO XMT1
 02E7 C3D503   D    491         JMP     CCCCC            ;JUMP TO CCCCC
 02EA 210000   E    492 XMT3:   LXI     H,CHAPTR         ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 02ED 56            493         MOV     D,M              ;MOVE CHAPTR-MSB INTO D-REG
 02EE 23            494         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 02EF 5E            495         MOV     E,M              ;MOVE CHAPTR-LSB INTO E-REG
 02F0 210000   E    496         LXI     H,ROUPTR         ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
 02F3 72            497         MOV     M,D              ;STORE CHAPTR-MSB IN ROUPTR-MSB
 02F4 23            498         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
 02F5 73            499         MOV     M,E              ;STORE CHAPTR-LSB IN ROUPTR-LSB
 02F6 111105   D    500         LXI     D,SPACE          ;LOAD ADDRESS OF SPACE INTO D-E-REG
 02F9 210000   E    501         LXI     H,CHAPTR         ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 02FC 72            502         MOV     M,D              ;MOVE MSB OF ADDRESS OF SPACE INTO CHAPTR-MSB
 02FD 23            503         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 02FE 73            504         MOV     M,E              ;MOVE LSB OF ADDRESS OF SPACE INTO CHAPTR-LSB
 02FF 210000   E    505         LXI     H,NOINST         ;LOAD ADDRESS OF NOINST INTO H-L-REG
 0302 3601          506         MVI     M,01H            ;SET NOINST TO 1
 0304 C3C003   D    507         JMP     SEND1            ;JUMP TO SEND1
                    508 ;
                    509 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    510 ;
                    511 ;      ROUTINE NAME:  NVECH2
                    512 ;      DESCRIPTION:  THIS ROUTINE CHECKS ECHOS FROM THE NOVA TO
                    513 ;                    SEE IF THEY MATCH THE CHARACTER SENT.
                    514 ;
                    515 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    516 ;
 0307 DB16          517 NVECH2: IN      16H              ;GET CHARACTER FROM NOVA THROUGH USART-3
 0309 E67F          518         ANI     7FH              ;RESET PARITY BIT
 030B 210000   E    519         LXI     H,CHAPTR         ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 030E 56            520         MOV     D,M              ;MOVE CHAPTR-MSB INTO D-REG
 030F 23            521         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 0310 5E            522         MOV     E,M              ;MOVE CHAPTR-LSB INTO E-REG
 0311 EB            523         XCHG                     ;SWAP CHAPTR INTO H-L-REG
 0312 BE            524         CMP     M                ;IS CHARACTER FROM NOVA EQUAL TO CHARACTER AT CHAPT
                        R?
 0313 CA2403   D    525         JZ      AAAAA            ;IF YES, JUMP TO AAAAA
 0316 210000   E    526         LXI     H,NOINST         ;LOAD ADDRESS OF NOINST INTO H-L-REG
 0319 3601          527         MVI     M,01H            ;SET NOINST TO 1
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE   10

  LOC  OBJ          LINE        SOURCE STATEMENT

  031B 3E7F          528         MVI     A,7FH            ;MOVE DELETE CHARACTER INTO A-REG
  031D F680          529         ORI     80H              ;SET PARITY BIT
  031F D316          530         OUT     16H              ;SEND DELETE CHARACTER TO NOVA THROUGH USART-3
  0321 C3E804   D    531         JMP     RETSYS           ;JUMP TO RETSYS
                     532 ;
                     533 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     534 ;
                     535 ;      ROUTINE NAME:  ABC
                     536 ;      DESCRIPTION:   THIS IS THE MAIN ROUTINE FOR FIELDING BAD CHARACTERS
                     537 ;                     TO PREVENT THEM FROM BEING SENT TO THE NOVA.  IT ALSO
                     538 ;                     WATCHES FOR THE END OF A BLOCK AND FINDS THE BEGINNING
                     539 ;                     OF THE NEXT BLOCK OF THE REPORT IF NECESSARY.
                     540 ;
                     541 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     542 ;
  0324 210000   E    543 AAAAA:  LXI     H,CHAAMT         ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
  0327 56            544         MOV     D,M              ;MOVE CHAAMT-MSB INTO D-REG
  0328 23            545         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
  0329 5E            546         MOV     E,M              ;MOVE CHAAMT-LSB INTO E-REG
  032A 1B            547         DCX     D                ;DECREMENT CHAAMT IN D-E-REG
  032B 73            548         MOV     M,E              ;UPDATE CHAAMT-LSB IN MEMORY
  032C 2B            549         DCX     H                ;DECREMENT H-L-REG TO ADDRESS OF CHAAMT-MSB
  032D 72            550         MOV     M,D              ;UPDATE CHAAMT-MSB IN MEMORY
  032E 210000   E    551         LXI     H,CHAPTR         ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  0331 56            552         MOV     D,M              ;MOVE CHAPTR-MSB INTO D-REG
  0332 23            553         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  0333 5E            554         MOV     E,M              ;MOVE CHAPTR-LSB TO E-REG
  0334 13            555         INX     D                ;INCREMENT CHAPTR IN D-E-REG
  0335 73            556         MOV     M,E              ;UPDATE CHAPTR-LSB IN MEMORY
  0336 2B            557         DCX     H                ;DECREMENT H-L-REG TO ADDRESS OF CHAPTR-MSB
  0337 72            558         MOV     M,D              ;UPDATE CHAPTR-MSB IN MEMORY
  0338 210000   E    559         LXI     H,CHAAMT         ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
  033B 3E00          560         MVI     A,00H            ;SET A-REG EQUAL TO ZERO
  033D BE            561         CMP     M                ;IS CHAAMT-MSB ZERO?
  033E C24603   D    562         JNZ     BBBBB            ;IF NO, JUMP TO BBBBB
  0341 23            563         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
  0342 BE            564         CMP     M                ;IS CHAAMT-LSB ZERO?
  0343 CAD503   D    565         JZ      CCCCC            ;IF YES, THEN JUMP TO CCCCC
  0346 210000   E    566 BBBBB:  LXI     H,CHAPTR         ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  0349 56            567         MOV     D,M              ;MOVE CHAPTR-MSB INTO D-REG
  034A 23            568         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  034B 5E            569         MOV     E,M              ;MOVE CHAFTR-LSB INTO E-REG
  034C EB            570         XCHG                     ;SWAP CHAPTR INTO H-L-REG
  034D 7E            571         MOV     A,M              ;MOVE CHARACTER AT CHAPTR INTO A-REG
  034E E67F          572         ANI     7FH              ;RESET PARITY BIT
  0350 FE0D          573         CPI     0DH              ;IS CHARACTER AT CHAPTR A CARRIAGE RETURN?
  0352 C27103   D    574         JNZ     BBB1             ;IF NO, JUMP TO BBB1
  0355 EB            575         XCHG                     ;SWAP CHAPTR BACK TO D-E-REG
  0356 13            576         INX     D                ;INCREMENT CHAPTR
  0357 73            577         MOV     M,E              ;UPDATE CHAPTR-LSB
  0358 2B            578         DCX     H                ;DECREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  0359 72            579         MOV     M,D              ;UPDATE CHAPTR-MSB
  035A 210000   E    580         LXI     H,CHAAMT         ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
  035D 56            581         MOV     D,M              ;MOVE CHAAMT-MSB INTO D-REG
  035E 23            582         INX     H                ;INCREMENT H-L-REG TO ADDRESS OF CHAAMT-LSB
  035F 5E            583         MOV     E,M              ;MOVE CHAAMT-LSB INTO E-REG
  0360 1B            584         DCX     D                ;DECREMENT CHAAMT IN D-E-REG
  0361 73            585         MOV     M,E              ;UPDATE CHAAMT-LSB IN MEMORY
  0362 2B            586         DCX     H                ;DECREMENT H-L-REG TO ADDRESS OF CHAAMT-MSB
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE   11

  LOC  OBJ          LINE        SOURCE STATEMENT

 0363 72             587         MOV     M,D             ;UPDATE CHAAMT-MSB IN MEMORY
 0364 210000    E    588         LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
 0367 3603           589         MVI     M,03H           ;SET NOINST TO 3
 0369 210000    E    590         LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 036C 3604           591         MVI     M,04H           ;SET NOVSTS TO 4 IN MEMORY
 036E C3E804    D    592         JMP     RETSYS          ;JUMP TO RETSYS
 0371 FE2C           593 BBB1:   CPI     2CH             ;IS CHARACTER A COMMA?
 0373 C28A03    D    594         JNZ     BBB3            ;IF NO, JUMP TO BBB3
 0376 CDC803    D    595         CALL    BBBSB1          ;MOVE CHAPTR INTO ROUPTR
 0379 111605    D    596         LXI     D,COMMA         ;LOAD ADDRESS OF COMMA INTO D-E-REG
 037C 210000    E    597 BBB2:   LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 037F 72             598         MOV     M,D             ;MOVE MSB OF ADDRESS OF COMMA TO CHAPTR-MSB
 0380 23             599         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 0381 73             600         MOV     M,E             ;MOVE LSB OF ADDRESS OF COMMA TO CHAPTR-LSB
 0382 210000    E    601         LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
 0385 3601           602         MVI     M,01H           ;SET NOINST TO 1
 0387 C3C003    D    603         JMP     SEND1           ;JUMP TO SEND1
 038A FE3E           604 BBB3:   CPI     3EH             ;IS CHARACTER AT CHAPTR A RIGHT ANGLE BRACKET?
 038C C29803    D    605         JNZ     BBB4            ;IF NO, JUMP TO BBB4
 038F CDC803    D    606         CALL    BBBSB1          ;MOVE CHAPTR INTO ROUPTR
 0392 211B05    D    607         LXI     H,ANGLE         ;LOAD ADDRESS OF ANGLE INTO H-L-REG
 0395 C37C03    D    608         JMP     BBB2            ;JUMP BACK TO BBB2
 0398 FE00           609 BBB4:   CPI     00H             ;IS CHARACTER AT CHAPTR A NULL?
 039A CA2403    D    610         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 039D FE05           611         CPI     05H             ;IS CHARACTER AT CHAPTR A CONTROL-E?
 039F CA2403    D    612         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 03A2 FE0A           613         CPI     0AH             ;IS CHARACTER AT CHAPTR A LINEFEED?
 03A4 CA2403    D    614         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 03A7 FE13           615         CPI     13H             ;IS CHARACTER AT CHAPTR AN X-OFF?
 03A9 CA2403    D    616         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 03AC FE1B           617         CPI     1BH             ;IS CHARACTER AT CHAPTR AN ESCAPE?
 03AE CA2403    D    618         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 03B1 FE5C           619         CPI     5CH             ;IS CHARACTER AT CHAPTR A BACKSLASH?
 03B3 CA2403    D    620         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 03B6 FE7F           621         CPI     7FH             ;IS CHARACTER AT CHAPTR A DELETE?
 03B8 CA2403    D    622         JZ      AAAAA           ;IF YES, JUMP TO AAAAA
 03BB 210000    E    623         LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
 03BE 3604           624         MVI     M,04H           ;SET NOINST TO 4
 03C0 210000    E    625 SEND1:  LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 03C3 3602           626         MVI     M,02H           ;SET NOVSTS TO 2
 03C5 C3E804    D    627         JMP     RETSYS          ;JUMP TO RETSYS
 03C8 210000    E    628 BBBSB1: LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 03CB 56             629         MOV     D,M             ;MOVE CHAPTR-MSB INTO D-REG
 03CC 23             630         INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 03CD 5E             631         MOV     E,M             ;MOVE CHAPTR-LSB INTO E-REG
 03CE 210000    E    632         LXI     H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
 03D1 72             633         MOV     M,D             ;MOVE CHAPTR-MSB INTO ROUPTR-MSB
 03D2 23             634         INX     H               ;INCREMENT H-L-REG TO ROUPTR-LSB
 03D3 73             635         MOV     M,E             ;MOVE CHAPTR-LSB INTO ROUPTR-LSB
 03D4 C9             636         RET                     ;RETURN TO INVOKING PROCEDURE
 03D5 210000    E    637 CCCCC:  LXI     H,CURBLK        ;LOAD ADDRESS OF CURBLK INTO H-L-REG
 03D8 7E             638         MOV     A,M             ;MOVE CURBLK TO A-REG
 03D9 C603           639         ADI     03H             ;ADD 3 TO GET MSB OF END OF BLOCK ADDRESS
 03DB 67             640         MOV     H,A             ;MOVE CURBLK-MSB + 3 TO H-REG
 03DC 2EFF           641         MVI     L,0FFH          ;MOVE FF INTO L-REG
 03DE 7E             642         MOV     A,M             ;MOVE LAST BYTE OF BLOCK TO A-REG
 03DF FE00           643         CPI     00H             ;IS LAST BYTE OF BLOCK ZERO?
 03E1 C20204    D    644         JNZ     CCC1            ;IF NO, JUMP TO CCC1
 03E4 210000    E    645         LXI     H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE   12

  LOC  OBJ          LINE        SOURCE STATEMENT

 03E7 3600           646         MVI    M,00H          ;SET ROUPTR-MSB TO ZERO
 03E9 23             647         INX    H              ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
 03EA 3600           648         MVI    M,00H          ;SET ROUPTR-LSB TO ZERO
 03EC 210000    E    649         LXI    H,CHAPTR       ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
 03EF 3600           650         MVI    M,00H          ;SET CHAPTR-MSB TO ZERO
 03F1 23             651         INX    H              ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 03F2 3600           652         MVI    M,00H          ;SET CHAPTR-LSB TO ZERO
 03F4 3E1B           653         MVI    A,1BH          ;MOVE ESCAPE CHARACTER INTO A-REG
 03F6 F680           654         ORI    80H            ;SET PARITY BIT
 03F8 D316           655         OUT    16H            ;SEND ESCAPE TO NOVA THROUGH USART-3
 03FA 210000    E    656         LXI    H,NOINST       ;LOAD ADDRESS OF NOINST INTO H-L-REG
 03FD 3605           657         MVI    M,05H          ;SET NOINST TO 5
 03FF C3E804    D    658         JMP    RETSYS         ;JUMP TO RETSYS
 0402 210000    E    659 CCC1:   LXI    H,CURBLK       ;LOAD ADDRESS OF CURBLK INTO H-L-REG
 0405 77             660         MOV    M,A            ;MOVE LAST BYTE TO LAST BLOCK TO CURBLK
 0406 210000    E    661         LXI    H,CHAPTR       ;LOAD ADDRESS OF CHAPTR INTO H-L-REG
 0409 77             662         MOV    M,A            ;MOVE CURBLK TO CHAPTR-MSB
 040A 23             663         INX    H              ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
 040B 3605           664         MVI    M,05H          ;SET CHAPTR-LSB TO 5
 040D 67             665         MOV    H,A            ;MOVE CURBLK TO H-REG
 040E 2E02           666         MVI    L,02H          ;MOVE 2 INTO L-REG
 0410 5E             667         MOV    E,M            ;MOVE CHAAMT-LSB INTO E-REG
 0411 23             668         INX    H              ;INCREMENT H-L-REG TO NEXT BYTE
 0412 56             669         MOV    D,M            ;MOVE CHAAMT-MSB INTO D-REG
 0413 210000    E    670         LXI    H,CHAAMT       ;LOAD ADDRESS OF CHAAMT-MSB INTO H-L-REG
 0416 7A             671         MOV    A,D            ;MOVE MSB INTO A-REG
 0417 FE03           672         CPI    03H            ;IS MSB A 3?
 0419 C22304    D    673         JNZ    CCC2           ;IF NO, JUMP TO CCC2
 041C 7B             674         MOV    A,E            ;MOVE LSB INTO A-REG
 041D FEFF           675         CPI    0FFH           ;IS LSB FF?
 041F C22304    D    676         JNZ    CCC2           ;IF NO, JUMP TO CCC2
 0422 1B             677         DCX    D              ;DECREMENT D-E-REG TO AVOID COUNTING LAST BYTE IN B
                         LOCK
 0423 1B             678 CCC2:   DCX    D              ;DECREMENT D-E-REG FOUR TIMES TO GET CHAAMT
 0424 1B             679         DCX    D
 0425 1B             680         DCX    D
 0426 1B             681         DCX    D
 0427 72             682         MOV    M,D            ;UPDATE CHAAMT-MSB
 0428 23             683         INX    H              ;INCREMENT H-L-REG TO CHAAMT-LSB
 0429 73             684         MOV    M,E            ;UPDATE CHAAMT-LSB
 042A C34603    D    685         JMP    BBBBB
                     686 ;
                     687 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     688 ;
                     689 ;      ROUTINE NAME:  NOVBYE
                     690 ;      DESCRIPTION:   THIS IS THE ROUTINE WHICH SETS UP 'BYE' FOR
                     691 ;                     TRANSMISSION TO THE NOVA.  IT ALSO ZEROS OUT
                     692 ;                     THE BLOCKS OF THE REPORT WHICH WAS JUST SENT
                     693 ;                     AND TURNS OFF THE TRANSMITTING BIT IN THE
                     694 ;                     USER'S STATUS BYTE.
                     695 ;
                     696 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                     697 ;
 042D DB16           698 NOVBYE: IN     16H            ;GET CHARACTER FROM NOVA THROUGH USART-3
 042F E67F           699         ANI    7FH            ;RESET PARITY BIT
 0431 FE52           700         CPI    52H            ;IS CHARACTER AN 'R'?
 0433 CAC904    D    701         JZ     EXIT           ;IF YES, JUMP TO EXIT
 0436 FE2A           702         CPI    2AH            ;IS CHARACTER AN ASTERISK?
 0438 CA4804    D    703         JZ     BYE1           ;IF YES, JUMP TO BYE1
```

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE  13

 LOC  OBJ          LINE        SOURCE STATEMENT

 043B 3E14          704         MVI    A,14H          ;SET TIME TO 20 MINUTES
 043D CD0000   E    705         CALL   TIMINT         ;INITIALIZE TIMER
 0440 210000   E    706         LXI    H,NOVSTS       ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
 0443 3601          707         MVI    M,01H          ;SET NOVSTS TO 1
 0445 C3E804   D    708         JMP    RETSYS         ;JUMP TO RETSYS
 0448 210000   E    709 BYE1:   LXI    H,USRFLG       ;LOAD ADDRESS OF USRFLG INTO H-L-REG
 044B 3EFF          710         MVI    A,0FFH         ;SET A-REG TO FF
 044D BE            711         CMP    M              ;IS USRFLG FF?
 044E CAA504   D    712         JZ     BYE5           ;IF YES, JUMP TO BYE5
 0451 7E            713         MOV    A,M            ;MOVE USRFLG TO A-REG
 0452 36FF          714         MVI    M,0FFH         ;SET USRFLG TO FF
 0454 210000   E    715         LXI    H,SNTP1        ;LOAD ADDRESS OF SNTP1 TO H-L-REG
 0457 3600          716         MVI    M,00H          ;SET SNTP1 TO ZERO
 0459 210000   E    717         LXI    H,SENT0        ;LOAD ADDRESS OF SENT0 INTO H-L-REG
 045C FE00          718         CPI    00H            ;IS USRFLG ZERO?
 045E CA6804   D    719         JZ     BYE2           ;IS YES, JUMP TO BYE2
 0461 24            720         INR    H              ;INCREMENT H-REG TO PRODUCE ADDRESS OF SENT1
 0462 FE01          721         CPI    01H            ;IS USRFLG 1?
 0464 CA6804   D    722         JZ     BYE2           ;IF YES, JUMP TO BYE2
 0467 24            723         INR    H              ;INCREMENT H-REG TO ADDRESS OF SENT2
 0468 46            724 BYE2:   MOV    B,M            ;MOVE SENTn INTO B-REG
 0469 04            725         INR    B              ;INCREMENT SENTn
 046A 70            726         MOV    M,B            ;UPDATE SENTn
 046B 210000   E    727         LXI    H,STAT         ;LOAD ADDRESS OF STAT0 INTO H-L-REG
 046E FE00          728         CPI    00H            ;IS USRFLG ZERO?
 0470 CA7A04   D    729         JZ     BYE3           ;IF YES, JUMP TO BYE3
 0473 24            730         INR    H              ;INCREMENT H-REG TO PRODUCE ADDRESS OF STAT1
 0474 FE01          731         CPI    01H            ;IS USRFLG 1?
 0476 CA7A04   D    732         JZ     BYE3           ;IF YES, JUMP TO BYE3
 0479 24            733         INR    H              ;INCREMENT H-REG TO PRODUCE ADDRESS OF STAT2
 047A 7E            734 BYE3:   MOV    A,M            ;MOVE STATn INTO A-REG
 047B E6F7          735         ANI    0F7H           ;RESET uTRANS BIT
 047D 77            736         MOV    M,A            ;UPDATE STATn
 047E 210000   E    737         LXI    H,FSTBLK       ;LOAD ADDRESS OF FSTBLK INTO H-L-REG
 0481 66            738         MOV    H,M            ;MOVE FSTBLK INTO H-REG
 0482 2E00          739 BYE4:   MVI    L,00H          ;SET L-REG TO ZERO
                    740         REPT   5
 -                  741         MVI    M,00H          ;ZERO THE BYTE
 -                  742         INX    H              ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
                    743         ENDM
 0484 3600          744+        MVI    M,00H          ;ZERO THE BYTE
 0486 23            745+        INX    H              ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
 0487 3600          746+        MVI    M,00H          ;ZERO THE BYTE
 0489 23            747+        INX    H              ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
 048A 3600          748+        MVI    M,00H          ;ZERO THE BYTE
 048C 23            749+        INX    H              ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
 048D 3600          750+        MVI    M,00H          ;ZERO THE BYTE
 048F 23            751+        INX    H              ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
 0490 3600          752+        MVI    M,00H          ;ZERO THE BYTE
 0492 23            753+        INX    H              ;INCREMENT H-L-REG TO ADDRESS OF NEXT BYTE
 0493 7C            754         MOV    A,H            ;MOVE MSB OF ADDRESS OF BEGINNING OF BLOCK TO A-REG
 0494 C603          755         ADI    03H            ;ADD THREE TO MSB OF ADDRESS OF BEGINNING OF BLOCK
 0496 67            756         MOV    H,A            ;MOVE MSB OF ADDRESS OF END OF BLOCK TO H-REG
 0497 2EFF          757         MVI    L,0FFH         ;SET LSB OF ADDRESS OF END OF BLOCK TO FF
 0499 7E            758         MOV    A,M            ;MOVE LAST BYTE OF BLOCK TO A-REG
 049A 3600          759         MVI    M,00H          ;ZERO THE LAST BYTE OF THE BLOCK
 049C FE00          760         CPI    00H            ;WAS LAST BYTE OF BLOCK ZERO?
 049E CAA504   D    761         JZ     BYE5           ;IF YES, JUMP TO BYE5
 04A1 67            762         MOV    H,A            ;MOVE LAST BYTE OF BLOCK INTO H-REG
```

```
  LOC  OBJ         LINE        SOURCE STATEMENT

  04A2 C38204   D   763          JMP     BYE4            ;JUMP BACK TO BYE4
  04A5 11FE04   D   764 BYE5:    LXI     D,BYE           ;LOAD ADDRESS OF BYE INTO D-E-REG
  04A8 210000   E   765          LXI     H,CHAPTR        ;LOAD ADDRESS OF CHAPTR-MSB INTO H-L-REG
  04AB 72           766          MOV     M,D             ;UPDATE CHAPTR-MSB
  04AC 23           767          INX     H               ;INCREMENT H-L-REG TO ADDRESS OF CHAPTR-LSB
  04AD 73           768          MOV     M,E             ;UPDATE CHAPTR-LSB
  04AE 11C904   D   769          LXI     D,EXIT          ;LOAD ADDRESS OF EXIT INTO D-E-REG
  04B1 210000   E   770          LXI     H,ROUPTR        ;LOAD ADDRESS OF ROUPTR-MSB INTO H-L-REG
  04B4 72           771          MOV     M,D             ;UPDATE ROUPTR-MSB
  04B5 23           772          INX     H               ;INCREMENT H-L-REG TO ADDRESS OF ROUPTR-LSB
  04B6 73           773          MOV     M,E             ;UPDATE ROUPTR-LSB
  04B7 3E14         774          MVI     A,14H           ;SET TIME FOR 20 MINUTES
  04B9 CD0000   E   775          CALL    TIMINT          ;INITIALIZE TIMER
  04BC 210000   E   776          LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
  04BF 3602         777          MVI     M,02H           ;SET NOVSTS TO 2
  04C1 210000   E   778          LXI     H,NOINST        ;LOAD ADDRESS OF NOINST INTO H-L-REG
  04C4 3601         779          MVI     M,01H           ;SET NOINST TO 1
  04C6 C3E804   D   780          JMP     RETSYS          ;JUMP TO RETSYS
                    781 ;
                    782 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    783 ;
                    784 ;     ROUTINE NAME:  EXIT
                    785 ;     DESCRIPTION:  THIS IS THE ROUTINE WHICH MASKS THE NOVA INTERRUPT
                    786 ;                   AND TURNS OFF THE DATA TERMINAL READY BIT AFTER
                    787 ;                   A REPORT HAS BEEN TRANSMITTED.
                    788 ;
                    789 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    790 ;
  04C9 210000   E   791 EXIT:    LXI     H,NOVSTS        ;LOAD ADDRESS OF NOVSTS INTO H-L-REG
  04CC 3605         792          MVI     M,05H           ;SET NOVSTS TO 5
  04CE DB19         793          IN      19H             ;READ CURRENT INTERRUPT STATUS REGISTER FROM PIC-0
  04D0 F680         794          ORI     80H             ;SET BIT FOR NOVA INTERRUPT MASK (INTERRUPT-6)
  04D2 D319         795          OUT     19H             ;PROGRAM PIC-0 TO MASK NOVA INTERRUPT
  04D4 3E25         796          MVI     A,25H           ;MOVE USART COMMAND WORD INTO A-REG
  04D6 D317         797          OUT     17H             ;TURN OFF DATA TERMINAL READY BIT IN USART-3
  04D8 210000   E   798          LXI     H,TIMCTR        ;LOAD ADDRESS OF TIMCTR INTO H-L-REG
  04DB 3601         799          MVI     M,01H           ;SET TIMCTR TO 1
  04DD 210000   E   800          LXI     H,TIMCT1        ;LOAD ADDRESS OF TIMCT1-MSB INTO H-L-REG
  04E0 3605         801          MVI     M,05H           ;SET TIMCT1-MSB TO 05
  04E2 23           802          INX     H               ;INCREMENT H-L-REG TO ADDRESS OF TIMCT1-LSB
  04E3 3600         803          MVI     M,00H           ;SET TIMCT1-LSB TO 00
  04E5 C3E804   D   804          JMP     RETSYS          ;JUMP TO RETSYS
                    805 ;
                    806 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    807 ;
                    808 ;     ROUTINE NAME:  RETSYS
                    809 ;     DESCRIPTION:  THIS IS THE ROUTINE WHICH RESTORES THE VALUES
                    810 ;                   OF ALL OF THE REGISTERS TO THEIR ORIGINAL STATE
                    811 ;                   AND RETURNS CONTROL TO THE INVOKING PROCEDURE.
                    812 ;
                    813 ;-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:-:
                    814 ;
  04E8 C1           815 RETSYS:  POP     B               ;RESTORE B-C-REG FROM STACK
  04E9 D1           816          POP     D               ;RESTORE D-E-REG FROM STACK
  04EA E1           817          POP     H               ;RESTORE H-L-REG FROM STACK
  04EB F1           818          POP     PSW             ;RESTORE A-REG AND FLAGS FROM STACK
  04EC C9           819          RET                     ;RETURN TO LOCATION AT TIME OF INTERRUPT
  04ED 555359       820 USY:     DB      'USY',0DH
  04F0 0D
```

EPA Form 2350-4 (4-80)

TYPING GUIDE SHEET

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        NOVA     PAGE   15

  LOC  OBJ         LINE        SOURCE STATEMENT

  04F1 45414459     821 EADY:   DB      'EADY',0DH
  04F5 0D
  04F6 43434F55     822 CCOU:   DB      'CCOU',0DH
  04FA 0D
  04FB 4158         823 AX:     DB      'AX',0DH
  04FD 0D
  04FE 425945       824 BYE:    DB      'BYE',0DH
  0501 0D
  0502 52554E22     825 RUN:    DB      'RUN"MICROINPUT',0DH
  0506 4D494352
  050A 4F494E50
  050E 5554
  0510 0D
  0511 3C33323E     826 SPACE:  DB      '<32>',0DH
  0515 0D
  0516 3C34343E     827 COMMA:  DB      '<44>',0DH
  051A 0D
  051B 3C36323E     828 ANGLE:  DB      '<62>',0DH
  051F 0D
                    829         END                     ;END OF MODULE

PUBLIC SYMBOLS
NOVA   D 0000

EXTERNAL SYMBOLS
BOTTOM E 0000   CHAAMT E 0000   CHAPTR E 0000   CURBLK E 0000   EOI534 E 0000   EOIM   E 0000
FSTBLK E 0000   NOINST E 0000   NOVSTS E 0000   ROUPTR E 0000   RUNFLG E 0000   SENT0  E 0000
SNTP1  E 0000   STAT   E 0000   STBY0  E 0000   SYSTAT E 0000   TIMCT1 E 0000   TIMCTR E 0000
TIMINT E 0000   TTYBUF E 0000   USRFLG E 0000

USER SYMBOLS
AAAAA  D  324   ACC1   D 0177   ACCID  D 0164   ANGLE  D 051B   AX     D 04FB   BBB1   D 0371
BBB2   D 037C   BBB3   D 038A   BBB4   D 0398   BBBBB  D 0346   BBBSB1 D 03C8   BOTTOM E 0000
BUSY   D 011D   BYE    D 04FE   BYE1   D 0448   BYE2   D 0468   BYE3   D 047A   BYE4   D 0482
BYE5   D 04A5   CCC1   D 0402   CCC2   D 0423   CCCCC  D 03D5   CCOU   D 04F6   CHAAMT E 0000
CHAPTR E 0000   COMMA  D 0516   CURBLK E 0000   EADY   D 04F1   ECH1   D 01B7   ECH2   D 01D0
ECH3   D 01E9   EOI534 E 0000   EOIM   E 0000   EXIT   D 04C9   FSTBLK E 0000   NOINST E 0000
NOVA   D 0000   NOVBYE D 042D   NOVECH D 0190   NOVESC D 012F   NOVSGN D 01FB   NOVSTS E 0000
NOVTST D 0029   NOVXMT D 0278   NVECH2 D 0307   ONLINE D 00F8   READY  D 012E   RETSYS D 04E8
ROUPTR E 0000   RUN    D 0502   RUNFLG E 0000   SEND1  D 03C0   SENT0  E 0000   SGN1   D 0220
SGN2   D 0252   SGN3   D 0268   SNTP1  E 0000   SPACE  D 0511   STAT   E 0000   STBY0  E 0000
SYSTAT E 0000   TIMCT1 E 0000   TIMCTR E 0000   TIMINT E 0000   TST1   D 0059   TST2   D 0078
TST3   D 0092   TST4   D 00AC   TST5   D 00C1   TSTSB1 D 00E5   TSTSB2 D 00E9   TTYBUF E 0000
USRFLG E 0000   USY    D 04ED   XMT1   D 028F   XMT2   D 02CE   XMT3   D 02EA

ASSEMBLY COMPLETE,   NO ERRORS
```

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                    PAGE   1

AAAAA    329     525     543#    610     612     614     616     618     620     622
ACC1     258     261     263#
ACCID     91     254#
ANGLE    607     828#
AX       103     823#
BBB1     574     593#
BBB2     597#    608
BBB3     594     604#
BBB4     605     609#
BBBBB    299     302     472     562     566#    685
BBBSB1   595     606     628#
BOTTOM    14
BUSY      71     100     195#
BYE      170     764     824#
BYE1     703     709#
BYE2     719     722     724#
BYE3     729     732     734#
BYE4     739#    763
BYE5     712     761     764#
CCC1     644     659#
CCC2     673     676     678#
CCCCC    444     491     565     637#
CCOU      94     822#
CHAAMT    14     228     296     439     478     543     559     580     670
CHAPTR    14      56      73      84      93     102     108     117     145     171     224     263     286     325
         330     379     445     492     501     519     551     566     597     628     649     661     765
COMMA    596     827#
CURBLK    14     637     659
EADY      85     821#
ECH1     292     304#
ECH2     306     317#
ECH3     318     330#
EOI534    14
EOIM      14
EXIT     701     769     791#
FSTBLK    14     737
NOINST    15      32     175     218     269     319     385     505     526     588     601     623     656     778
NOVA      12      17      28#
NOVBYE    45     698#
NOVECH    37     284#
NOVESC   165     215#
NOVSGN    39     357#
NOVSTS    15     177     234     267     309     338     387     590     625     706     776     791
NOVTST    35      56#
NOVXMT    41     434#
NVECH2    43     517#
ONLINE    78     165#
READY     82     214#
RETSYS    46      76      87      96      98     105     116     129     182     202     242     274     295     303
         340     362     373     415     419     437     531     592     627     658     708     780     804     815#
ROUPTR    15      70      81      90      99     130     149     166     220     311     321     496     632     645
         770
RUN      378     825#
RUNFLG    15     383
SEND1    507     603     625#
SENTO     15     717
SGN1     360     374#
SGN2     393     396     398#
SGN3     406     409     411#
SNTP1     15     412     715
SPACE    500     826#
```

BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD: BEGIN SECTIONS HERE
BEGIN LAST LINE OF TEXT
7/8"
OF IMAGE AREA: OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
PAGE NUMBER
EPA Form 2350-4 (4-80)
(PREVIOUSLY CIN. EPA FORM 287)
U.S. GOVERNMENT PRINTING OFFICE
TYPING GUIDE SHEET

```
ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE, V2.1                 PAGE   2

STAT        15    391    727
STBY0       15    404
SYSTAT      15    197    366    374
TIMCT1      16    800
TIMCTR      16    798
TIMINT      16    196    233    308    372    705    775
TST1        69     77#
TST2        80     88#
TST3        89     97#
TST4        60     65    106#
TST5       114    117#
TSTSB1      72     75     83     86     92     95    101    104    140#
TSTSB2     115    134    144#
TTYBUF      16    256
USRFLG      16    254    389    709
USY         74    820#
XMT1       441    445#   488    490
XMT2       453    455    457    459    461    463    465    467    469    473#
XMT3       471    492#

CROSS REFERENCE COMPLETE
```

EPA Form 2350-4 (4-80)
TYPING GUIDE SHEET

SECTION 8

NOTES TO THE SYSTEM MANAGER

8.1 CONNECTING THE MICROCOMPUTER

The microcomputers used by the Microprocessor Buffer System have five I/O ports. The system console must be connected to the port on the iSBC 80/20-4 central processing unit board. Up to three intelligent instrument interface boxes can be connected to ports 0, 1, and 2 on the iSBC 534 I/O expansion board. Port 3 of that board must be connected to a BASIC user's port on the NOVA 840.

The system console must be a Hewlett-Packard 2640 Series CRT. This terminal should be set as follows:

Duplex: Full
Parity: None
Baud Rate: 1200

8.2 BOOTING THE SYSTEM

To boot the system, turn on the microcomputer and the system console. Type

```
UUUU
```

on the console. The system will respond with the message

```
80/20 MONITOR V1.1
```

and it will type a period to prompt for a command. Type the command

```
G800)
```

and the Microprocessor Buffer System will begin the series of prompts shown in Figure 22.

For ports 0, 1, and 2, you must specify the baud rate, the five starting characters, the five stopping characters, and the account ID. You can select 110, 300, or 1200 baud as the rate at which the system will receive characters from the intelligent instruments. However, it is important to choose a balance of baud rates which will minimize the chances of a memory overflow. If all three of the instruments transmit to the microprocessor at 110 vaud, and the microprocessor relays the data to the NOVA at 1200 baud,

EPA Form 2350-4 (4-80)
TYPING GUIDE SHEET

the microprocessor's memory will never overflow (unless the NOVA stops responding, of course). With the instruments transmitting at 300 baud and the NOVA receiving at 1200 baud, the memory should not overflow unless all of the instruments deliver several large reports in a row with no dead time between them. If the microcomputer is receiving from three instruments at 1200 baud, and transmitting to the NOVA at 1200 baud, the microcomputer will soon overflow its memory unless the reports are separated by long dead times.

The five starting characters and five stopping characters are used by the system to recognize when an intelligent instrument is transmitting data that should be relayed to the NOVA. The five starting characters must be a string which occurs at the very beginning of every report, but which is never printed by the intelligent instrument between reports. The five stopping characters must be a string which is always included in the last line of a report, but never in any other line of a report.

For example, the Varian 220L Chromatography Data System prints the letters "GC" indented two spaces in the upper left corner of each report, and the last line of the report begins with the words "TOTAL AREA." Therefore, the five starting characters "b̸b̸GCb̸" are used to identify the beginning of reports, and the five stopping characters "ALb̸AR" tell the system to stop collecting data at the end of the line which contains this string. Another example of using this feature involves the Perkin Elmer Sigma 10 Chromatography Data System which does not provide any consistant starting or stopping characters in its reports. Therefore, a BASIC program was written in the Sigma 10 to print the word "START" before each report and the string "/STOP" after each report.

The account ID for each of the first three ports is simply the ID the system should use to log onto the NOVA.

8.3 THE SYSTEM STATUS REPORT ON THE SYSTEM CONSOLE

When all of the initial questions have been answered, the system fills its console screen as shown in Figure 23, and begins to update this report about every 1.6 seconds. For each of the three users, this report shows whether the system is ready to receive a report, receiving a report, standing by with a complete report, and/or transmitting a report. These indicators correspond to the lights on the interface boxes in the laboratories. The system can never be both ready to receive and receiving for the same user, but all other combinations of asterisks are quite possible.

The console report will usually include the line "SYSTEM RUNNING NORMALLY," but that phrase may be replaced by one of the error messages "ALL LINES BUSY OR MAX USERS," "ID IN USE OR UNKNOWN," "NOVA NOT RESPONDING," or "MEMORY OVERFLOW." If any of the first three messages appears, the red "ERROR" lights on the interface boxes will come on, but the system will continue to receive data. If the "MEMORY OVERFLOW" message appears, the "STOP DATA" lights on the interface boxes will come on, and the system will stop receiving data from the instruments, although it will continue to send data to the NOVA, if possible.

EPA Form 2350-4 (4-80)

TYPING GUIDE SHEET

```
           ENVIRONMENTAL MONITORING AND SUPPORT LABORATORY - CINCINNATI
                      ADVANCED INSTRUMENTATION SECTION

                        MICROPROCESSOR BUFFER SYSTEM
                                VERSION  2.1
PORT 0.....

       SPEEDS:    1 = 110 BAUD
                  2 = 300 BAUD
                  3 =1200 BAUD

     ENTER SPEED (1, 2, OR 3): 2

          PORT SET TO 300 BAUD

     ENTER FIVE STARTING CHARACTERS: ABCDE

     ENTER FIVE STOPPING CHARACTERS: VWXYZ

     ENTER ACCOUNT ID: MIKE
```

FIGURE 22. BOOTING PROCEDURE PROMPTS.

```
           ENVIRONMENTAL MONITORING AND SUPPORT LABORATORY - CINCINNATI
                      ADVANCED INSTRUMENTATION SECTION

                        MICROPROCESSOR BUFFER SYSTEM
                                VERSION  2.1

         USER 0    USER 1    USER 2

uP READY    *         *         *

uP RCVG                                SYSTEM RUNNING NORMALLY

uP STBY                                BLOCKS IN USE:  00

uP TRANS                               BUFFER SIZE:  48 BLOCKS
```

FIGURE 23. SYSTEM CONSOLE STATUS INFORMATION.

EPA-287 (Cin.)
(4-76)

The "BLOCKS IN USE" counter is also updated regularly in the status report on the system console.

8.4 NOVA DIRECTORIES, LOGIN MESSAGES, AND BASIC.ID REQUIREMENTS

The following is a sample of how BASIC.ID should be configured in the NOVA for the use of the Microprocessor Buffer System:

```
USR1/DZ0:DIRECTORY/LOGINMSG1
USR2/DZ0:DIRECTORY/LOGINMSG1
MIC1/DZ0:DIRECTORY/LOGINMSG2
```

In this example, there is one shared directory, namely DZ0:DIRECTORY. There are three account IDs, namely USR1, USR2, and MIC1. USR1 and USR2 are account IDs for human users. LOGINMSG1 is the login message for human users; it probably gives some message of the day and then chains directly to an automation system. MIC1 is the account ID used by the Microprocessor Buffer System when it logs onto the NOVA. LOGINMSG2 is a special login message program which must consist of only the following two statements (and perhaps some remarks):

```
10 ESC
20 NEW
```

Because the directory is shared, it must be INITed from the background CLI and never RELEASEd.

8.5 THE BASIC PROGRAM IN THE NOVA

When the Microprocessor Buffer System logs on the NOVA, it issues the BASIC command RUN"MICROINPUT). You must supply a program named MICROINPUT which will receive the strings from the microcomputer and file them appropriately. Figure 24 is the listing of a very rudimentary program of this type which simply prints the report on the lineprinter. There are several important principles exemplified in this program. First, the program prompts with a question mark when it is ready for another string. Second, as the program receives data from the microcomputer, it places it in a temporary file (the ARRIVAL file in this example). Later, the essential information can be gleaned from this temporary file and compacted into a more efficient file for the automation system to use. Third, the ON ESC THEN GOTO condition is in effect while the program is receiving data from the microcomputer. After the last line of the report has been transmitted, the Microprocessor Buffer System sends an escape to the NOVA, and this ON ESC THEN GOTO statement causes control to be transferred to the next section of the program. Fourth, as far as is practical, all possible errors have been anticipated, and error handling routines have been provided for them. In this example, the only such routine is the one which covers the situation of the lineprinter being in use when it is needed. In actual applications, the BASIC program must receive the entire report from the microcomputer, even if it cannot decipher it, or the Microprocessor Buffer System will continue to send the same report again and again.

TYPING GUIDE SHEET
BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
DROPPED HEAD, BEGIN SECTIONS HERE

```
0010 REM     [][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][]
0020 REM     []                                                             []
0030 REM     []     PROGRAM NAME:  MICROINPUT                               []
0040 REM     []     DESCRIPTION:   RECEIVES DATA FROM MICROPROCESSOR        []
0050 REM     []                    BUFFER SYSTEM AND PRINTS IT ON THE       []
0060 REM     []                    LINEPRINTER                              []
0070 REM     []     BY:  J. KOPKE (UCCC)                                    []
0080 REM     []     MOST RECENT REVISION:  SEPTEMBER 15, 1980               []
0090 REM     []                                                             []
0100 REM     [][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][][]
0110 REM
0120 DIM A$[132]
0130 PAGE =132
0140 ON ERR THEN GOTO 0400
0150 REM       ------------------------------------------------------------
0160 REM             RECEIVE LINES OF REPORT FROM MICRO BUFFER SYSTEM
0170 ON ESC THEN GOTO 0240
0180 OPEN FILE[1,1],"ARRIVAL"
0190 INPUT "?",A$
0200 WRITE FILE[1],A$
0210 GOTO 0190
0220 REM       ------------------------------------------------------------
0230 REM             PRINT LINES OF REPORT ON THE LINEPRINTER
0240 ON ESC THEN STOP
0250 CLOSE FILE[1]
0260 OPEN FILE[1,3],"ARRIVAL"
0270 GOSUB 0470
0280 PRINT FILE[2],"     D-A-T-A   F-R-O-M   M-I-C-R-O   B-U-F-F-E-R   S-Y-S-T-E-M"
0290 PRINT FILE[2]
0300 LREAD FILE[1],A$
0310 IF EOF(1) THEN GOTO 0340
0320 PRINT FILE[2],A$
0330 GOTO 0300
0340 CLOSE
0350 DELETE "ARRIVAL"
0360 ON ERR THEN STOP
0370 STOP
0380 REM       ------------------------------------------------------------
0390 REM             ERROR HANDLING ROUTINE
0400 ON ERR THEN STOP
0410 GOSUB 0470
0420 PRINT FILE[2],"FATAL ERROR CAUSED BY DATA FROM MICRO BUFFER SYSTEM"
0430 CLOSE FILE[2]
0440 STOP
0450 REM       ------------------------------------------------------------
0460 REM             SUBROUTINE TO OPEN LINEPRINTER
0470 ON ERR THEN GOTO 0510
0480 OPEN FILE[2,1],"$LPT"
0490 ON ERR THEN GOTO 0400
0500 RETURN
0510 IF SYS(7)<>-48 THEN GOTO 0400
0520 DELAY =10
0530 GOTO 0470
```

FIGURE 24.
RUDIMENTARY MICROINPUT PROGRAM.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA; OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS
3/8"
PAGE NUMBER
EPA-287 (Cin.)
(4-76)

For an example of an actual MICROINPUT Program, see the documentation for the program GC21 in the document, "Program Documentation for the Gas Chromatography Automation System."[2]

APPENDIX A

JUMPER CONFIGURATIONS FOR THE iSBC 80/20-4 SINGLE BOARD COMPUTER

Three changes have been made to the Single Board Computer. The Microprocessor Buffer System requires more ROM memory than the board allows as it is delivered from the factory. The following changes are required to jumper the board for a total of 8K for EPROM: Capacitors C35, C53, and C72 are removed. These jumpers are changed:

REMOVE	INSTALL
W2,A-C	W2,A-B
W4,B-D	W4,A-D
W4,C-E	W4,B-E
W7,A-B	W7,A-D
W8,A-C	W8,A-B

On the interrupt jumper pad, a jumper is installed from pin 28 to pin 47 to link the interrupt line from the iSBC 534 board to the INT4 interrupt request line.

Also, a jumper is soldered from pin 3 to pin 1 of chip A17 and then to pin 13 of chip A17. This links the RTS, CTS, DSR lines for the system console so that the computer will work with any terminal.

All other jumpers are left as installed at the factory. Jumper configurations are discussed in detail in the Intel Corporation publication _System 80/20-4 Microcomputer Hardware Reference Manual_, Chapter 2, "Preparation for Use."

EPA Form 2350-4 (4-80)

TYPING GUIDE SHEET

APPENDIX B

JUMPER CONFIGURATIONS FOR THE iSBC 534 FOUR CHANNEL I/O EXPANSION BOARD

The Microprocessor Buffer System requires several changes to the jumpers on the iSBC 534 Board. The I/O base address is hexidecimal 10. This is set by removing the jumper between pins 118 and 125, and replacing it with a jumper from pin 124 to pin 125.

Timer sections 1 and 2 (outputs BDG4 and BDG5) of Programmable Interval Timer A36 (PIT1) are connected in series. This option is selected by removing the jumper between pins 105 and 106, and replacing it with a jumper between pins 104 and 105.

The factory-installed jumpers for interrupts (pin 132 to 140, and pin 131 to 140) are removed, and INTR1 from PICO is connected to INT4 on the multibus by the installation of a jumper from pin 131 to pin 137.

The serial I/O port interface for all four ports is RS232C. This option is jumpered at the factory by the installation of 18-pin DIP header jumpers in socket W1 for port 0, socket W2 for port 1, socket W3 for port 2, and socket W4 for port 3. However, the DIP header jumperrs have been modified with soldered connections linking the pins for RTS, CTS, and DSR, since these signals are not provided by the processors connected to the ports.

All other jumper selectable options are left as installed at the factory. Jumper configurations are discussed in detail in the Intel Corporation publication *iSBC 534 Four Channel Communications Expansion Board Hardware Reference Manual*, Chapter 2, "Preparation for Use."

EPA Form 2350-4 (4-80)
TYPING GUIDE SHEET

Appendix C

VARIABLES LOCATED IN UTILITY STORAGE ON THE iSBC 80/20-4 BOARD

VARIABLE	ADDRESS	USER	DESCRIPTION
BLK0	3C12	User 0	Most significant byte of the address of the memory block currently in use by the user.
BLK1	3D12	User 1	
BLK2	3E12	User 2	
BLKP00	3C13	User 0	The least significant byte of the pointer which points to the memory position where the next arriving character is to be stored. The pointer is the offset from BLKn, and not the absolute address.
BLKP10	3D13	User 1	
BLKP20	3E13	User 2	
BLKP01	3C14	User 0	The most significant byte of the pointer above.
BLKP11	3D14	User 1	
BLKP21	3E14	User 2	
CHAAMT	3F31 - 3F32	System	The number of characters yet to be sent to the NOVA.
CHAPTR	3F13 - 3F14	System	The address of the next character to be sent to the NOVA.
CURBLK	3F1F	System	The most significant byte of the address of the current memory block from which a report is being transmitted to the NOVA.
FSTBLK	3F30	System	The most significant byte of the address of the first memory block of the report which is currently being transmitted to the NOVA.
LSTUSR	3F15	System	The number of the last user to have sent a report to the NOVA.
NOINST	3F16	System	The flag for which of the six routines in module NOVA should be executed when the next interrupt occurs.

EPA Form 2350-4 (4-80)
TYPING GUIDE SHEET

Appendix C (Continued)

VARIABLE	ADDRESS	USER	DESCRIPTION
NOVSTS	3F17	System	The flag for which of the six routines in module NOVAST should be executed the next time NOVAST gets control of the computer.
OVRFLW	3F11	System	A flag indicating that an overflow condition has been detected and the subroutine FRESH has assumed the duties normally carried out by the WAIT module.
ROUPTR	3F18 - 3F19	System	The address of the next routine to be executed.
RUNFLG	3F33	System	A flag indicating that the routine CWECHO is currently sending the string RUN"MICROINPUT to the NOVA.
SENT0	3C18	User 0	A counter for the number of reports sent to the NOVA for this user. This counter flips back to 00 after FF.
SENT1	3D18	User 1	
SENT2	3E18	User 2	
SEQ0	3C11	User 0	A counter used to keep track of the number of starting characters of stopping characters that have been received.
SEQ1	3D11	User 1	
SEQ2	3E11	User 2	
SNTP1	3F1A	System	The STBY value of the next report to be sent for a user.
STAT0	3C10	User 0	The User Status Word.
STAT1	3D10	User 1	
STAT2	3E10	User 2	
STBY0	3C15	User 0	The counter for the number of reports received by the Microcomputer Buffer System for this user. This counter flips back to 00 after FF.
STBY1	3D15	User 1	
STBY2	3E15	User 2	
SYSTAT	3F10	System	The System Status Word.
TIMCT1	3F1C - 3F1D	System	The counter which is decremented each time NOVAST is called to keep track of the passage of time. Decrementing TIMCT1 from $1C00_H$ to 0_H takes one minute.
TIMCTR	3F1B	System	The number of minutes of time delay which are allowed to elapse before the system gives up on the current operation.

Appendix C (Continued)

VARIABLE	ADDRESS	USER	DESCRIPTION
TOP	3F20	System	The most significant byte of the address of the top of memory.
TTYBUF0	3CB0 - 3CBE	User 0	The five starting characters, five stopping characters, and account ID for the user.
TTYBUF1	3DB0 - 3DBE	User 1	
TTYBUF2	3EB0 - 3EBE	User 2	
USRFLG	3F1E	System	The number of the user for whom a report is currently being sent to the NOVA.
VECFLG	3F34	System	A flag to indicate that the OPER routine should print null characters instead of finishing the interrupted console update.

BEGIN FIRST LINE OF TEXT HERE
CENTER OF PAGE
TOP OF IMAGE AREA
SHIFTED HEAD BEGIN SECTIONS HERE

Appendix D

MICROPROCESSOR MEMORY ALLOCATION

MEDIUM	ADDRESSES	CONTENTS
EPROM on 80/20-4 Board	0000 to 106AE	80/20 Monitor
	06AF to 07FF	not used
	0800 to 1C8E	Microprocessor Buffer System code
	1C8F to 1FFF	not used
no physical memory	2000 to 2FFF	not used
	3000 to 37FF	not used
RAM on 80/20-4 Board	3800 to 3A0B	Stack
	3A0C to 3BFF	not used
	3C00 to 3FDF	Microprocessor Buffer System utility storage
	3FE0 to 3FFF	Interrupt Jump Table
RAM on Memory Expansion Boards	4000 to 43FF	First Memory Block
	4400 to 47FF	Second Memory Block
	4800 to 4BFF	Third Memory Block
	. . .	. . .
	All addresses are given in hexidecimal notation.	If there is 16K of RAM on expansion boards, there are 16 Memory Blocks, the last one ending at 7FFF. If there is 32K of RAM on expansion boards, there are 32 Memory Blocks, the last one ending at BFFF. If there is 48K of RAM on expansion boards, there are 48 Memory Blocks, the last one ending at FFFF.

BEGIN LAST LINE OF TEXT
BOTTOM OF IMAGE AREA OUTSIDE DIMENSION FOR TABLES AND ILLUSTRATIONS

PAGE NUMBER
EPA Form 2350-4 (4-80)
(PREVIOUSLY CIN. EPA FORM 257)
TYPING GUIDE SHEET

REFERENCES

1. Budde, W. L., B. P. Almich, and J. M. Teuschler. The Status of the EPA Laboratory Automation Project. EPA-600/4-77-025, U.S. Environmental Protection Agency, Cincinnati, Ohio, 1977.

2. Kopke, Jonathan E., Program Documentation for the Gas Chromatography Automation System. EPA-600/4-80-011, U.S. Environmental Protection Agency, Cincinnati, Ohio, 1980.

EPA Form 2350-4 (4-80)
(PREVIOUSLY C/N. EPA FORM 287)
TYPING GUIDE SHEET